МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное
учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

Институт **ИВТИ**
Кафедра **УИТ**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(бакалаврская работа)

Направление **27.03.04 Управление в технических системах**
(код и наименование)

Направленность (профиль) **Управление и информатика в технических системах**

Форма обучения **очная**
(очная/очно-заочная/заочная)

Тема: **Исследование подходов к выявлению веб-атак, основанных на HTTP-запросах**

Студент **А-01-19** _____ **Антипов Д.А.**
группа подпись фамилия и инициалы

Научный руководитель **к.т.н.** **доцент** _____ **Мохов А.С.**
уч. степень должность подпись фамилия и инициалы

Консультант _____
уч. степень должность подпись фамилия и инициалы

Консультант _____
уч. степень должность подпись фамилия и инициалы

«Работа допущена к защите»

Зав. кафедрой **д.т.н.** **доцент** _____ **Бобряков А.В.**
уч. степень звание подпись фамилия и инициалы

Дата _____

Москва, 2023

МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

Институт **ИВТИ**
Кафедра **УИТ**

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
(бакалаврскую работу)

Направление **27.03.04 Управление в технических системах**
(код и наименование)

Направленность (профиль)

Управление и информатика в технических системах

Форма обучения **очная**
(очная/очно-заочная/заочная)

Тема: **Исследование подходов к выявлению веб-атак, основанных на HTTP-запросах**

Студент	**А-01-19**			**Антипов Д.А.**
	группа		подпись	фамилия и инициалы
Научный руководитель	**к.т.н.**	**доцент**		**Мохов А.С.**
	уч. степень	должность	подпись	фамилия и инициалы
Консультант				
	уч. степень	должность	подпись	фамилия и инициалы
Консультант				
	уч. степень	должность	подпись	фамилия и инициалы
Зав. кафедрой	**д.т.н.**	**доцент**		**Бобряков А.В.**
	уч. степень	звание	подпись	фамилия и инициалы

Место выполнения работы **Кафедра управления и интеллектуальных технологий**

СОДЕРЖАНИЕ РАЗДЕЛОВ ЗАДАНИЯ И ИСХОДНЫЕ ДАННЫЕ

Введение

1. Виды сетевых атак.
   1.1. Актуальность задачи выявления сетевых атак
   1.2. HTTP-запросы
   1.3. Этапы атаки
   1.4. Средства и методы обеспечения безопасности
   1.5. Обзор видов сетевых атак
2. Подходы к выявлению веб-атак с помощью машинного обучения
   2.1. Задача классификации
   2.2. Способы предварительной обработки
3. Проведение исследований по выявлению и классификации веб-атак на основе http-запросов
   3.1. Описание выборки
   3.2. Применение выбранных подходов к решению поставленной задачи
   3.3. Проверка результатов на другом наборе данных

Исходные данные – набор веб-запросов

URL: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/OGOIXX

ПЕРЕЧЕНЬ ГРАФИЧЕСКОГО МАТЕРИАЛА

Количество листов -

Количество слайдов в презентации 12

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Tomás Sureda Riera, Juan-Ramón Bermejo Higuera. A new multi-label dataset for Web attacks CAPEC classification using machine learning techniques: [Электронный ресурс]. URL: https://www.sciencedirect.com/science/article/pii/S0167404822001833
1. H. Gouk, B. Pfahringer, M. Cree. Learning distance metrics for multi-label classification: [Электронный ресурс]. URL: https://proceedings.mlr.press/v63/Gouk8.html
2. G. Madjarov, D. Kocev, D. Gjorgjevikj, S. Džeroski. An extensive experimental comparison of methods for multi-label learning: [Электронный ресурс]. URL: https://www.sciencedirect.com/science/article/abs/pii/S0031320312001203

АННОТАЦИЯ

Целью данной выпускной квалификационной работы является исследование подходов к выявлению веб-атак, основанных на HTTP-запросах. В ходе исследований предполагается обнаруживать веб-атаки на основе машинного обучения. В качестве исходных данных использовался набор данных SR-BH 2020 с несколькими метками.

В результате были предложены два способа предварительной обработки данных для последующей классификации веб-запросов.

Выпускная квалификационная работа состоит из: введения, трех глав, разделенных на параграфы, заключения и списка литературы.

В первой главе была рассмотрена актуальность и проблематика данной задачи, затронуты распространенные методы обнаружения веб-атак. Также в данной главе приведены основные виды сетевых атак, используемых в процессе исследований.

Вторая глава посвящена подходам к выявлению веб-атак с помощью машинного обучения, а также указаны особенности предварительной обработки данных, представляющих из себя веб-запросы.

Третья глава включает описание используемого набора данных и выбранных подходов к решению поставленной задачи.

В заключении приведены основные выводы, полученные в результате проведенных исследований.

Общий объем работы – 71 страница – содержит 8 иллюстраций, 12 таблиц, 10 источников литературы.

Ключевые слова: веб-атака, веб-запрос, классификация, набор данных, сайт, приложение, уязвимость, угроза.

Оглавление

ВВЕДЕНИЕ

Веб-атаки на основе HTTP-запросов — это популярный вид атак, который используется для злоумышленного манипулирования или эксплуатации веб-запросов, отправляемых клиентами к веб-приложениям или серверам. Злоумышленники могут использовать различные методы и уязвимости для осуществления таких атак.

Распознавание веб-атак имеет огромное значение в области кибербезопасности по нескольким причинам:

- распознавание веб-атак позволяет обнаружить их на ранних стадиях, что дает возможность быстро реагировать и принимать меры по их предотвращению. Чем быстрее атака будет распознана, тем быстрее можно предпринять действия для остановки атакующего и минимизации ущерба.
- многие веб-атаки направлены на кражу или компрометацию данных. Распознавание этих атак позволяет быстро установить факт нарушения и принять меры для защиты конфиденциальной информации и предотвращения дальнейших утечек данных.
- успешные веб-атаки могут привести к нарушению репутации организации или бренда. Быстрое распознавание атак и принятие мер по их предотвращению помогает минимизировать негативные последствия и сохранить доверие пользователей и клиентов и др.

В целом, распознавание веб-атак является важным компонентом стратегии кибербезопасности и помогает обеспечить защиту веб-приложений, серверов и данных пользователя от вредоносных действий злоумышленников.

В данной работе будут рассмотрены подходы к решению задачи распознавания и классификации веб-атак, основанных на HTTP-запросах.

Глава 1. ВИДЫ СЕТЕВЫХ АТАК

1.1. Актуальность выявления сетевых атак

Сегодня Интернет стал активным центром деловой активности, и множество компаний непосредственно зависят от глобальных информационных технологий. Большое количество предприятий и организаций во всем мире используют компьютерные системы для управления своими производственными процессами, управления персоналом, распределения ресурсов и обеспечения удаленного доступа для пользователей и других важных задач.

Использование информационных технологий принесло ряд явных преимуществ, таких как снижение затрат, ускорение производственных процессов, увеличение мобильности и быстрого доступа к информации и услугам. Это также привело к появлению новых рынков услуг, таких как удаленное управление банковскими счетами, онлайн-заказы и оплата товаров и услуг через Интернет. В результате стоимость информации, которая циркулирует в корпоративных и глобальных сетях, значительно возросла. Но с развитием информационных технологий наблюдается также резкий рост компьютерной преступности. Необходимо учитывать, что успешное функционирование информационной инфраструктуры предприятий, их конкурентоспособность, доходы и позиция на рынке в значительной мере зависят от надежности и целостности их информационных ресурсов, а также защиты конфиденциальной информации от несанкционированного доступа. В связи с этим возрастают требования к системам защиты от сетевых атак. Они должны обеспечивать не только пассивную блокировку несанкционированного доступа к внутренним ресурсам предприятия из внешних сетей, но и обнаружение успешных атак, анализ причин возникновения угроз информационной безопасности и, по возможности, автоматическое устранение этих угроз.

С каждым годом значительно увеличивается количество атак на веб-серверы и приложения; платформы электронной коммерции, финансовые и правительственные учреждения, крупные корпорации и т.д. подвергаются веб-атакам по экономическим или идеологическим причинам. По данным DDoS-Guard число DDoS-атак на инфраструктуру за прошлый год превысило показатели 2021 года более чем на 700%, достигнув 1,2 млн инцидентов. Наибольшее количество атак, около 1 млн, произошло на уровне приложений (Application Layer или L7 - имитируют обращение пользовательского приложения, например просмотр веб-страниц). DDoS-Guard прогнозирует, что к весне 2023 года количество DDoS-атак увеличится на 300% по сравнению с концом 2022 года. При этом урон, наносимый этими атаками, будет значительно больше, так как злоумышленники сосредоточатся на качестве атак, а не только на их количестве. Дмитрий Никонов, руководитель направления защиты на уровне приложений в компании, отмечает, что хакеры будут активно использовать уязвимости и осуществлять локальные атаки на ключевые сервисы, которые широко используются множеством организаций [1].

Кроме того, простые и эффективные методы атак на веб-сайты, такие как атаки с внедрением SQL (SQLi) и межсайтового скриптинга (XSS), являются серьезной угрозой. По данным специализированной страховой компании Hiscox, воздействие кибератак, которым подверглись компании, в 17% случаев представляло угрозу для их жизнеспособности. Интересно отметить, что в 29% случаев веб-сайт становился первой точкой входа для атаки [2].

1.2. HTTP-запросы

Для лучшего понимания различных типов веб-атак и их причин необходимо обозначить основные функции и операции веб-приложений. Веб-приложение представляет собой программу, которая использует веб-браузеры и веб-технологии для выполнения задач посредством Интернета. Для обработки запросов от клиента веб-приложению требуется веб-сервер, а для выполнения запрошенных задач - сервер приложений. Информация, необходимая для работы приложения, обычно хранится в базе данных.

Практически все веб- и мобильные приложения, которыми мы пользуемся, постоянно обмениваются данными с глобальной сетью. Большинство таких обменов осуществляются с использованием HTTP-запросов в качестве протокола связи. Злоумышленники используют уязвимости и недостатки в протоколе HTTP с целью получения несанкционированного доступа к системе или проведения других вредоносных действий. Такие методы называются атаками на основе HTTP-запросов.

HTTP-запрос (Hypertext Transfer Protocol request) - это метод, с помощью которого клиент (обычно веб-браузер) отправляет запрос на сервер для получения определенного ресурса или выполнения определенного действия. Протокол HTTP является основным протоколом для передачи данных в Интернете и обеспечивает взаимодействие между клиентом и сервером. HTTP-протокол описывается в стандарте RFC 2616 [3]. В настоящее время наиболее широко распространена версия протокола HTTP/2, но все еще можно встретить использование более ранней версии - HTTP/1.1. Обмен информацией по протоколу HTTP включает в себя взаимодействие между клиентом и сервером. На рисунке 1.1 представлен процесс коммуникации устройств по HTTP-протоколу.

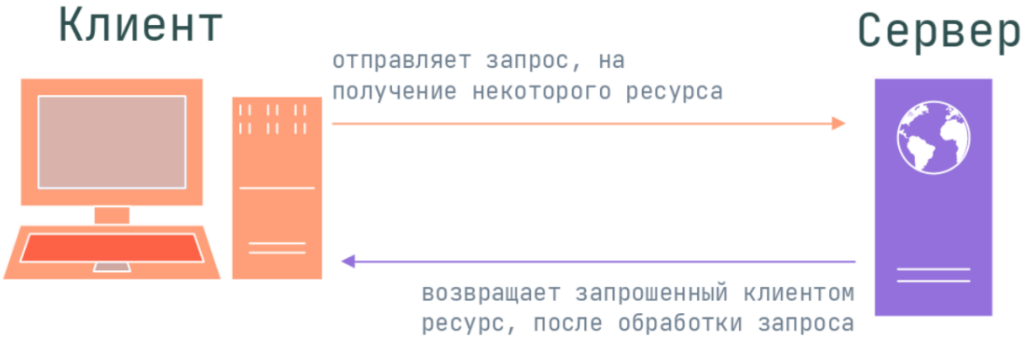

Рисунок 1.1 – Схема обмена информацией по HTTP-протоколу

Передача данных между клиентом и сервером в рамках протокола HTTP осуществляется с использованием сообщений. Эти сообщения бывают двух типов:

– Запросы (HTTP Requests): клиент отправляет запросы на сервер, чтобы запросить выполнение определенных действий или получить доступ к определенному ресурсу. Запрос состоит из HTTP-заголовка, который содержит информацию о действии, которое клиент хочет выполнить.
– Ответы (HTTP Responses): ответ содержит информацию, которую сервер передает обратно клиенту в ответ на его запрос.

Каждое сообщение представляет собой текстовую информацию, разделенную на несколько строк.

HTTP-запрос состоит из нескольких частей:

– Метод: Определяет тип запроса, который клиент хочет выполнить на сервере. Некоторые распространенные методы включают GET, POST, PUT, DELETE и HEAD. Например, метод GET используется для получения информации с сервера, а метод POST - для отправки данных на сервер.
– URL (Uniform Resource Locator): Указывает адрес ресурса, к которому клиент хочет получить доступ. URL состоит из протокола

(например, http://), доменного имени и пути к конкретному ресурсу на сервере.

- Версия используемого протокола: определяет структуру следующих за стартовой строкой данных.
- Заголовки: представляют собой метаданные запроса, которые содержат информацию о клиенте, типе контента, языке и других параметрах запроса. Заголовки помогают серверу понять, как обрабатывать запрос и возвращать соответствующий ответ.
- Тело запроса: не все HTTP-методы предполагают наличие тела. Например, методы GET, HEAD, DELETE, OPTIONS обычно не требуют наличия тела в запросе. Однако некоторые типы запросов могут содержать данные, которые отправляются на сервер в теле запроса. Самым распространенным методом, который используется для отправки данных, является метод POST.

После того, как клиент отправляет HTTP-запрос, сервер обрабатывает запрос и возвращает HTTP-ответ, который содержит статус запроса, заголовки и, возможно, тело ответа с запрошенными данными или результатами выполненного действия.

1.3. Этапы атаки

Процесс веб-атаки на веб-приложение включает несколько этапов, каждый из которых направлен на нарушение безопасности приложения и получение несанкционированного доступа или нанесение вреда. Рассмотрим подробнее этапы веб-атаки на веб-приложение:

1. **Разведка**: на этом этапе злоумышленник собирает информацию о целевом веб-приложении. Он может использовать различные методы, включая анализ исходного кода приложения, сканирование портов и обнаружение служб, сбор информации о сервере и его конфигурации, а также изучение доступных страниц и функциональности приложения.

Целью разведки является выявление уязвимостей и слабых мест в приложении, которые могут быть использованы для дальнейшей атаки.

2. **Поиск уязвимостей**: после завершения разведки злоумышленник ищет конкретные уязвимости в веб-приложении, которые могут быть эксплуатированы. Это может включать поиск уязвимостей в коде приложения, ошибки конфигурации сервера, незащищенные точки входа или слабо защищенные аутентификационные механизмы. Целью этого этапа является определение точки входа для атаки и выбор метода атаки, наиболее подходящего для обнаруженных уязвимостей.
3. **Эксплуатация уязвимостей**: на этом этапе злоумышленник использует обнаруженные уязвимости для выполнения злонамеренных действий в приложении. Например, если найдена уязвимость в системе аутентификации, злоумышленник может использовать ее для обхода авторизации и получения доступа к ограниченным ресурсам. Если обнаружена SQL-инъекция, злоумышленник может выполнить вредоносный SQL-код для извлечения, изменения или удаления данных из базы данных. Другие распространенные методы эксплуатации уязвимостей включают межсайтовый скриптинг (XSS), межсайтовую подделку запроса (CSRF) и выполнение удаленного кода (RCE).
4. **Пост-эксплуатация:** повторное и дальнейшее использование обнаруженных уязвимостей, если эксплуатация уязвимостей прошла успешно.

1.4. Средства и методы обеспечения безопасности

Сейчас разработчики программного обеспечения активно развивают технологии обнаружения компьютерных атак. Они внедряют в свои продукты инструменты, которые позволяют обнаруживать и анализировать

изменения, происходящие в информационных системах. Решения становятся более сложными и комбинируют различные инструменты.

Во многих организациях используются только базовые средства защиты, которые, чаще всего, не достаточно эффективны для обнаружения сложных целенаправленных атак и полноценного анализа событий. На рисунке 1.2 наглядно представлен процесс веб-атаки.

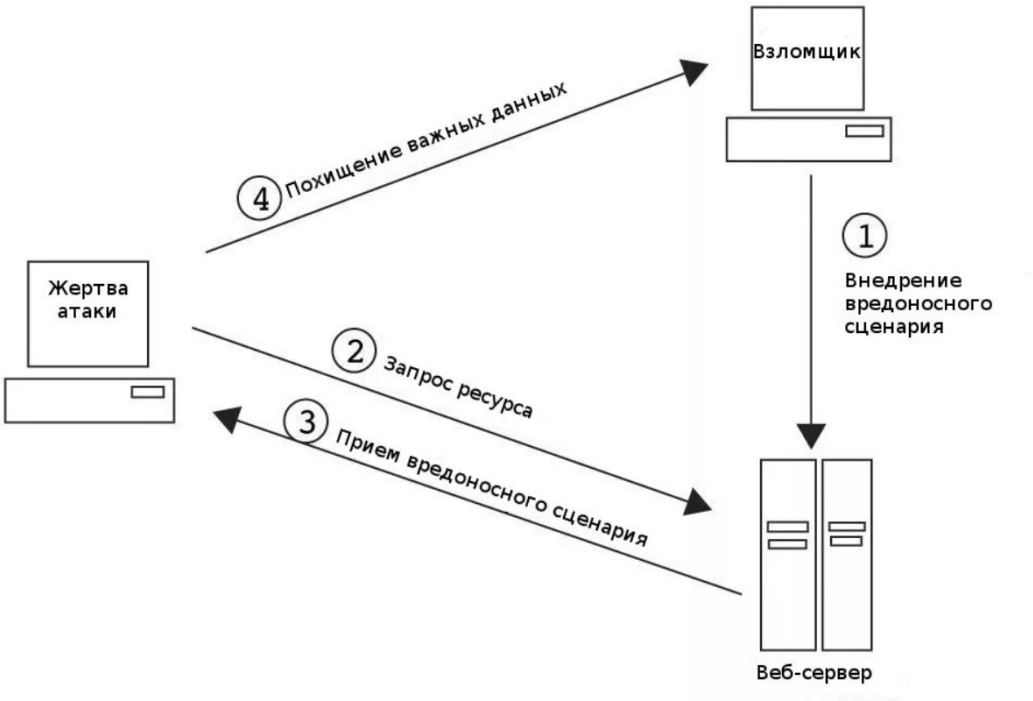

Рисунок 1.2 – процесс веб-атаки на веб-приложение

В настоящее время существует несколько методов и технологий для обнаружения веб-атак.

Анализ журналов сервера

Обнаружение веб-атак может осуществляться путем анализа журналов сервера. Журналы содержат информацию о входящих запросах, ошибочных запросах, аутентификации и других событиях. Анализ этих журналов позволяет выявить подозрительную активность, необычные запросы или аномалии в поведении.

Мониторинг сетевого трафика

Мониторинг сетевого трафика позволяет обнаруживать аномальную активность и потенциальные атаки на веб-приложение. С помощью систем обнаружения вторжений (СОВ) или специализированных инструментов можно анализировать трафик, искать необычные запросы, подозрительные пакеты данных или атаки типа отказа в обслуживании (DoS) и распределенных атак отказа в обслуживании (DDoS).

Сканеры уязвимостей

Сканеры уязвимостей позволяют автоматически сканировать веб-приложение на наличие известных уязвимостей. Эти инструменты отправляют тестовые запросы, анализируют ответы и ищут признаки уязвимостей, такие как SQL-инъекции, межсайтовый скриптинг или уязвимости аутентификации. Уязвимостные сканеры помогают выявить потенциальные проблемы безопасности и указывают на необходимость их исправления.

Системы обнаружения веб-атак (WAF)

Одним из самых популярных способов защиты веб-приложений является использование веб-приложений брандмауэра (Web Application Firewall, WAF). WAF размещается между веб-приложением и клиентом, где он анализирует входящий трафик на прикладном уровне модели OSI, ищет сигнатуры атак и блокирует вредоносный трафик. Основная задача WAF заключается в сопоставлении входящего трафика с базой данных сигнатур или правил, поэтому обновление этой базы данных является критически важным. Например, если обнаруживается необычно большое количество запросов от одного IP-адреса или необычные параметры запросов, WAF может считать это подозрительным и предпринять соответствующие действия.

Преимущество WAF состоит в том, что он независим от языка программирования веб-приложения, поскольку он действует до того, как вредоносный трафик будет выполнять код. WAF обновляется регулярно с новыми сигнатурами и образцами атак, чтобы обнаруживать и блокировать известные уязвимости. Это позволяет защитить веб-приложение от новых и широко распространенных видов атак. WAF может анализировать содержимое HTTP-запросов и ответов, включая заголовки, параметры, тело запроса и тело ответа. Она может искать специфические шаблоны или ключевые слова, которые могут указывать на потенциальную вредоносную активность или уязвимости [4].

Существуют несколько способов обойти защиту, обеспечиваемую WAF. Некоторые из них включают: использование нестандартных HTTP-заголовков (злоумышленники могут создать HTTP-запросы с нестандартными заголовками, которые могут обойти обнаружение WAF; это может включать изменение или маскировку значений заголовков, чтобы обойти фильтрацию), изменение кодировки данных (злоумышленники могут изменять кодировку данных, передаваемых веб-приложению, чтобы обойти обнаружение WAF; они могут использовать URL-кодировку, шестнадцатеричное кодирование или другие методы для скрытия вредоносных данных), обход детекции шаблонов (WAF может основываться на обнаружении определенных шаблонов атак; злоумышленники могут изменять свои атакующие запросы, чтобы избежать обнаружения по сигнатурам или правилам, используемым WAF), Slow HTTP-атаки (это тип атаки, при котором злоумышленник отправляет медленные или фрагментированные HTTP-запросы, чтобы перегрузить сервер и обойти обнаружение WAF, которое может быть ориентировано на обнаружение более явных форм атак).

RASP (Runtime Application Self-Protection)

Это подход в области безопасности веб-приложений, который предлагает защиту приложений непосредственно во время их выполнения (в режиме реального времени). RASP внедряет защитные механизмы в само приложение или его окружение, обеспечивая более глубокий уровень защиты и обнаружения атак. Основная идея RASP заключается в том, что защитные механизмы непосредственно встроены в исполняющую среду или код приложения и отслеживают его поведение в реальном времени. Это позволяет выявлять и предотвращать атаки на самом уровне выполнения кода, когда доступны подробные данные о поведении приложения и его окружении.

Преимущества RASP включают: более глубокая и точечная защита, так как он работает непосредственно на уровне выполнения кода; динамическая адаптация к изменяющимся условиям и атакам, основываясь на данных, полученных в режиме реального времени: интегрируется непосредственно в приложение и может быть настроен для минимального влияния на его производительность. Однако, как и у любой технологии, у RASP есть свои ограничения. Некоторые из них включают ограниченную поддержку языков программирования и зависимость от правильной настройки и интеграции с приложением. В целом, RASP является эффективным инструментом для обеспечения безопасности веб-приложений, позволяющим обнаруживать и предотвращать атаки на уровне выполнения кода в режиме реального времени [5].

Системы мониторинга событий безопасности (SIEM)

Решения, входящие в класс Security Information and Event Management (SIEM), предназначены для отслеживания событий, которые происходят в различных информационных системах и приложениях. Эти решения включают в себя следующие возможности и функции:

– сбор и анализ больших объемов событий безопасности

- мониторинг текущего состояния средств защиты IT-инфраструктуры
- обнаружение компьютерных инцидентов в режиме реального времени
- получение полной картины происходящего в IT-инфраструктуре
- обнаружение и реагирование на сбои в работе IT- и ИБ-систем
- построение карты сети для прогнозирования цепочек атак
- получение данных для анализа и оценки риска в реальном времени
- выполнение отдельных требований и нормативных актов законодательства Российской Федерации в сфере мониторинга событий информационной безопасности

Злоумышленники часто успешно проникают и устанавливаются в инфраструктуре, оставаясь незамеченными на протяжении длительного времени. Их целью является скрытая компрометация данных или подготовка и осуществление внутренних атак на критические узлы инфраструктуры, избегая обнаружения администраторами и специалистами по информационной безопасности. Для более эффективного обнаружения таких инцидентов и борьбы с проникновениями в систему, необходимо использовать решения класса SIEM, которые позволяют проводить анализ событий в ретроспективе с использованием актуальной информации о угрозах, таких как фиды, индикаторы компрометации и экспертные правила корреляции и другие.

Принцип работы решений класса SIEM заключается в сборе логов (событий) от различных устройств на уровне программного обеспечения и аппаратных компонентов [6]. Все события стандартизируются в единый формат для последующего анализа. Корреляция событий, связанных с одним и тем же элементом инфраструктуры, может указывать на возможную кибератаку.

SIEM позволяет получить более полную картину происходящего в ИТ-инфраструктуре и анализировать сетевую связность различных узлов. Благодаря тщательному анализу и корреляции данных из разных источников,

SIEM способен обнаружить инциденты, которые могут остаться незамеченными при использовании отдельных средств обнаружения.

Системы анализа сетевого трафика (NTA)

Решения, относящиеся к классу Network Traffic Analysis (NTA), предназначены для обнаружения сетевых атак, перехвата и анализа сетевого трафика. Эти системы помогают выявить наличие злоумышленников на ранних этапах атаки, оперативно локализовать угрозы и обеспечить соблюдение стандартов информационной безопасности.

В отличие от стандартных сетевых анализаторов (IDS/IPS), NTA-системы анализируют трафик не только на периметре сети, но и внутри информационно-технической инфраструктуры. Этот класс решений имеет возможность сохранять сессии сетевого трафика, что позволяет создавать доказательную базу и передавать ее правоохранительным органам и в Государственную систему обнаружения, предупреждения и ликвидации последствий компьютерных атак на информационные ресурсы Российской Федерации. Кроме того, NTA-системы могут проводить анализ сетевого трафика, хранящегося в архиве, с использованием актуальных сигнатур (ретроспективный анализ). Решения класса NTA могут служить дополнительным источником сетевых событий для систем класса SIEM в случаях обнаружения сложных целенаправленных атак.

На практике такие решения позволяют, например, обнаружить подозрительные попытки подключения с неавторизованных узлов к контроллеру домена. Затем можно проанализировать исторические данные о сетевой активности данного узла и проверить, были ли ранее схожие попытки. Если такие попытки были зарегистрированы, это может указывать на целенаправленную атаку или попытки взлома.

Средства обнаружения компьютерных атак на конечных устройствах (EDR)

Решения класса Endpoint Detection and Response (EDR) обеспечивают обнаружение компьютерных атак на конечных устройствах и предоставляют необходимые метрики для реагирования специалистов по информационной безопасности [7].

В данном классе решений обычно используется специальный агент, который устанавливается на конечные устройства. Его задача включает сбор информации о действиях пользователей и программного обеспечения, обнаружение признаков компрометации (Indicators of compromise, IoC), помощь в выявлении и локализации скомпрометированных устройств и другие функции. Собранная информация играет важную роль в расследовании компьютерных инцидентов.

Решение EDR может включать различные технологии обнаружения в зависимости от его конкретной реализации. Например, это может быть агент для сбора и анализа данных, инструмент антивирусной защиты с поведенческим анализом, анализаторы индикаторов компрометации, а также возможность автоматического взаимодействия с SIEM-системами и системами класса Threat Intelligence для обогащения информацией об угрозах. Вместе с другими системами они предоставляют специалистам по информационной безопасности более полное представление о произошедшем и позволяют провести качественное расследование.

Применение решений класса EDR позволяет организациям обнаруживать сложные угрозы, которые направлены на обход традиционных средств защиты на конечных устройствах. Например, если злоумышленник пытается получить доступ к целевой системе, используя фишинговые рассылки или программные закладки, службе безопасности необходимо обладать средствами обнаружения и контроля, которые позволят получить оперативные данные для немедленного принятия мер по блокированию атаки. Если на конечных устройствах (серверах и рабочих станциях)

установлен агент EDR, который анализирует изменения в системе в реальном времени, то при активации вредоносной программы, которая открывает порты и начинает передачу данных на внешние ресурсы или шифрует жесткие диски, EDR обнаружит эти действия и передаст информации.

Система анализа сетевого трафика нового поколения (NDR)

Совсем недавно исследовательский центр Gartner сформировал новый класс решений под названием NDR (Network Detection & Response). Решения NDR включают в себя следующие технологии:

- анализ сетевого трафика
- поведенческая аналитика (машинное обучение для оперативного обнаружения, расследования и реагирования на угрозы)

В основе NDR лежат технологии NTA, дополняемые историческими метаданными, которые позволяют проводить расследование и поиск угроз, а также автоматически реагировать на угрозы путем интеграции с различными инструментами управления IT и ИБ, такими как межсетевые экраны и средства управления доступом к сети.

Решения NDR обеспечивают полную видимость в сети с возможностью обнаружения угроз в режиме реального времени. Интеграция с продуктами EDR и SIEM позволяет более точно коррелировать данные для выявления инцидентов.

Системы учета и обработки компьютерных угроз (Threat Intelligence)

Решения класса Threat Intelligence позволяют осуществлять детальный анализ уязвимостей и угроз в IT-инфраструктуре. Анализ проводится на основе информации о конкретных угрозах или посредством Threat Intelligence Feeds (TIF) - наборов данных, содержащих индикаторы компрометации (IoC), которые позволяют идентифицировать потенциальные угрозы. Например, организация может использовать эти данные для блокировки запросов с подозрительных IP-адресов. Однако, у TIF есть

недостаток - отсутствие контекста, связывающего эти индикаторы с конкретными атаками и вредоносными файлами. Для получения полной картины требуется проводить подробный анализ связей между данными, что обычно требует значительных усилий и часто выполняется вручную. В таких ситуациях специалистам приходят на помощь Threat Intelligence Platforms (TI-платформы).

TI-платформы - это специализированные платформы, предназначенные для обогащения, обнаружения, распространения и корреляции информации об угрозах. Эти решения автоматически связывают TIF, дополняя их контекстной информацией, что позволяет получить максимальную пользу от индикаторов компрометации. Одним из значительных преимуществ TI-платформ является возможность интеграции и централизованной обработки данных из различных источников информации об угрозах, а также интеграция с другими инструментами в области кибербезопасности, например, с SIEM-системами, платформами для реагирования на инциденты или средствами защиты.

На практике такие решения помогают аналитикам находить следы компрометации в сети и системах, используя определенные признаки. Например, если NTA-решение обнаруживает нелегитимный трафик, такой как SYN-flood атака (нападение типа отказа в обслуживании, при котором отправляется большое количество запросов на подключение по протоколу TCP в короткий промежуток времени), оно отправляет отчет в SIEM-систему, где данные анализируются, и формируется инцидент с информацией об «IP-адресе» и вердиктом - SYN-flood. Далее SIEM передает эту информацию в TI-платформу для дополнительного анализа. TI-платформа начинает проверять свою базу данных на наличие похожих DNS-записей. Если в наборе компрометационных индикаторов обнаруживается несколько десятков других IP-адресов, то эти адреса проверяются и передаются аналитикам. Аналитики передают полученную информацию администраторам ИТ-инфраструктуры, которые вносят соответствующие

изменения в настройки маршрутизаторов или межсетевых экранов, предотвращая потенциальные нелегитимные подключения заранее.

Поиск угроз (Threat hunting)

Процесс Threat hunting или «охота на угрозы» заключается в активном поиске следов компрометации или признаков вредоносной активности с целью обнаружения и нейтрализации угрозы [8]. Он отличается от реактивного режима работы средств обнаружения и защиты, поскольку аналитик активно исследует различные гипотезы, чтобы обнаружить следы вторжения. Главная цель заключается в том, чтобы обнаружить угрозу раньше, чем злоумышленники смогут ею воспользоваться. Для успешной «охоты на угрозы» крайне важно, чтобы аналитики получали телеметрию в режиме реального времени, и в этом им помогают описанные выше системы SIEM/EDR/NDR. Основные составляющие процесса threat hunting:

1. **Сбор данных**
   - информация от конечных устройств (процессы, запущенные на устройстве, файлы, хранимые на дисках)
   - данные сети (сетевые соединения, работа DNS-серверов, маршрутизаторов, коммутаторов и т.д.)
   - информация по сработкам из SIEM, NTA/NDR, данные из TI-платформы (это могут быть внешние базы фидов, либо собственная база, накапливаемая по результатам работы центра реагирования), также данные об IT-инфраструктуре организации (конфигурации, используемое ПО, уязвимости и т.п.).
2. **Аналитика** – начиная со сбора статистики, заканчивая поведенческим анализом на основе машинного обучения (User and Entity Behavioral Analytics, UEBA).
3. **Исследование полученных данных.**

4. **Нейтрализация атаки и разработка сценария реагирования на такого рода атаки.**

Проверка гипотез осуществляется с использованием опыта специалистов из центра реагирования и информации, получаемой из различных указанных выше источников. Одним из важных компонентов для поиска угроз является собственная база знаний центра реагирования, которая накапливает данные о расследованиях и результатах проверки предыдущих гипотез.

Средства поведенческого анализа («песочницы»)

Средство поведенческого анализа, известное также как «песочница», представляет собой изолированную среду, предназначенную для безопасного запуска исполняемых файлов. В такой среде обычно ограничивается доступ к сети и считыванию информации с устройств ввода, либо эти действия частично эмулируются.

Песочницы используются для анализа файлов на предмет наличия вредоносного программного обеспечения (ВПО) или для запуска подозрительного кода. Они моделируют среду, в которой предполагается запуск проверяемого объекта, что позволяет экспертам проводить оперативный поведенческий анализ и своевременно выявлять потенциальные угрозы.

Технология поведенческого анализа файлов на рынке существует уже довольно давно. В настоящее время методы этой технологии совершенствуются путем интеграции с другими классами решений по обнаружению и защите, а также благодаря появлению более гибких и настраиваемых сред, что обеспечивает более комплексный анализ опасностей.

На практике «песочницы» широко применяются. Основное назначение состоит в поведенческом анализе подозрительных файлов в окружении, максимально имитирующем реальную IT-инфраструктуру.

Приманки для хакеров («Honeypot»)

Решения класса «Honeypot» разработаны для обнаружения попыток взлома и изучения используемых методов, чтобы прогнозировать атаки и принимать меры по их противодействию. Они используют «приманки» - изолированные среды с явными уязвимостями, открытыми портами и другими недостатками, отделенные от основных промышленных систем. Целевыми объектами могут быть веб-серверы, виртуальные машины, сетевые устройства и другие системы и сервисы, обычно используемые организацией. Внутри среды, предназначенной для обнаружения атакующих, размещаются различные ресурсы, замаскированные под важные файлы, почтовые сообщения или данные учетных записей, чтобы привлечь внимание киберпреступников.

Кроме того, для борьбы с фишингом применяются «ханипот-ссылки», которые встроены в код веб-сайтов и позволяют обнаружить случаи клонирования веб-сайтов, предупреждая заранее защищающихся об возможной атаке.

Технология постоянно развивается, и появляются различные инструменты для раскрытия не только используемых методов, но и действий злоумышленников после их подключения к собственным системам, например, путем записи в терминальном окне. Это позволяет получить больше информации о самих атакующих и отследить, к каким другим системам они подключаются.

Для захвата более сложных злоумышленников развиваются «ханипоты нового поколения», которые могут виртуализировать целые сети.

1.5. Обзор видов сетевых атак.

История развития веб-атак на протяжении последних десятилетий отражает постоянное совершенствование техник злоумышленников и адаптацию к изменяющейся среде информационных технологий.

- 1990-е годы: В начале 1990-х годов с появлением World Wide Web (WWW) и распространением веб-сайтов стали появляться и первые веб-атаки. В это время основной акцент делался на эксплуатацию уязвимостей серверов, таких как отказ в обслуживании (DoS) и удаленное выполнение кода (RCE). Однако, веб-атаки были еще в основном простыми и неструктурированными.
- 2000-е годы: В этом десятилетии наблюдалось значительное увеличение сложности и масштабов веб-атак. Злоумышленники начали активно использовать уязвимости веб-приложений, такие как SQL-инъекции (SQLi) и межсайтовый скриптинг (XSS). Такие атаки позволяли злоумышленникам взламывать сайты, получать доступ к базам данных и манипулировать содержимым веб-страниц. Также появились атаки на сеансовую аутентификацию и фишинговые атаки, направленные на обман пользователей для получения их личной информации.
- 2010-е годы: В этом десятилетии веб-атаки продолжили развиваться и становиться более сложными и целенаправленными. Появилось больше продвинутых методов атаки, таких как атаки на основе алгоритмов машинного обучения, распределенные атаки отказа в обслуживании (DDoS), атаки на API, а также атаки на мобильные приложения. Защита от веб-атак стала более сложной, и компании стали активно принимать меры для обеспечения безопасности своих веб-приложений.

На сегодняшний день веб-атаки продолжают эволюционировать, и злоумышленники постоянно ищут новые уязвимости и методы атаки.

Для понимания различных аспектов безопасности информации, прежде всего, необходимо понимать разницу между такими понятиями, как «угроза» и «уязвимость».

Угроза (threat) - это потенциальная возможность наступления события, которое может привести к ущербу для информационной системы или организации. Угрозы могут быть представлены злоумышленниками, естественными явлениями или даже неосторожными действиями пользователей. Угрозы могут использовать уязвимости для проведения атаки или компрометации системы и данных. Примеры угроз включают вредоносное программное обеспечение, фишинг, атаки отказом в обслуживании (DDoS), кражу данных и другие.

Угрозы в информационных системах могут быть 3 типов:

- нарушения целостности
- нарушения доступности
- нарушения конфиденциальности

Уязвимость (vulnerability) - это слабое место или недостаток в системе, программном обеспечении, сети или процессе, которые могут быть использованы злоумышленниками для несанкционированного доступа, атаки или компрометации информации. Уязвимости могут быть обусловлены ошибками в проектировании, реализации, конфигурации или использовании системы. Примеры уязвимостей включают открытые порты, слабые пароли, уязвимые коды программ, недостаточную защиту данных и другие.

Таким образом, уязвимость представляет собой существующую слабость или недостаток, который может быть использован для атаки, в то время как угроза - это потенциальная возможность злоумышленника или события, которая может эксплуатировать уязвимость и причинить ущерб системе или организации. Уязвимости являются основой для угроз и служат основой для принятия мер по обеспечению информационной безопасности.

Большое количество уязвимостей приводит к их огромному разнообразию, а степень опасности каждой уязвимости может значительно различаться. Поэтому для эффективного определения приоритетов и первоочередного устранения самых опасных уязвимостей необходимо правильно классифицировать и оценивать их. Существуют различные международные и локальные системы классификации, принятые в разных странах. Эти системы включают списки известных уязвимостей в компьютерных системах, которые формируют экспертные группы, состоящие из специалистов по безопасности, разработчиков и заинтересованных представителей компаний в отрасли. Разберем подробнее систему классификации CAPEC, которая используется в данном исследовании.

Классификация шаблонов компьютерных атак CAPEC

В процессе развития совокупностей уязвимостей и недостатков безопасности программного обеспечения появилась классификация, называемая CAPEC (Common Attack Pattern Enumeration and Classification). CAPEC также описывает и классифицирует шаблоны атак, то есть общие элементы и методы, применяемые при атаках на уязвимые компоненты [9]. CAPEC расширяется через обсуждения на официальном форуме и почтовые рассылки. Эта классификация представляет собой еще один инструмент для систематизации информации о типичных атаках и их характеристиках, обеспечивая более полное понимание таких сценариев атак и помогая в разработке соответствующих контрмер.

В CAPEC применяется аналогичный иерархический подход. В рамках этой классификации были разработаны две основные модели (механизмы атак и объекты атак), а также несколько вспомогательных. В модели «механизмы атак» шаблоны атак иерархически упорядочены в соответствии с часто используемыми механизмами для эксплуатации уязвимостей. Категории, такие как «внедрение непредвиденных элементов», в этой модели

отражают различные методы, применяемые для атаки на систему, но не уточняют цели и последствия атаки. В модели «объекты атак» категории содержат описания компонентов, на которые направлена атака, например, «передача данных».

На рисунке 1.3 ниже приведен граф связей CAPEC-112 (Brute Force) в разных представлениях.

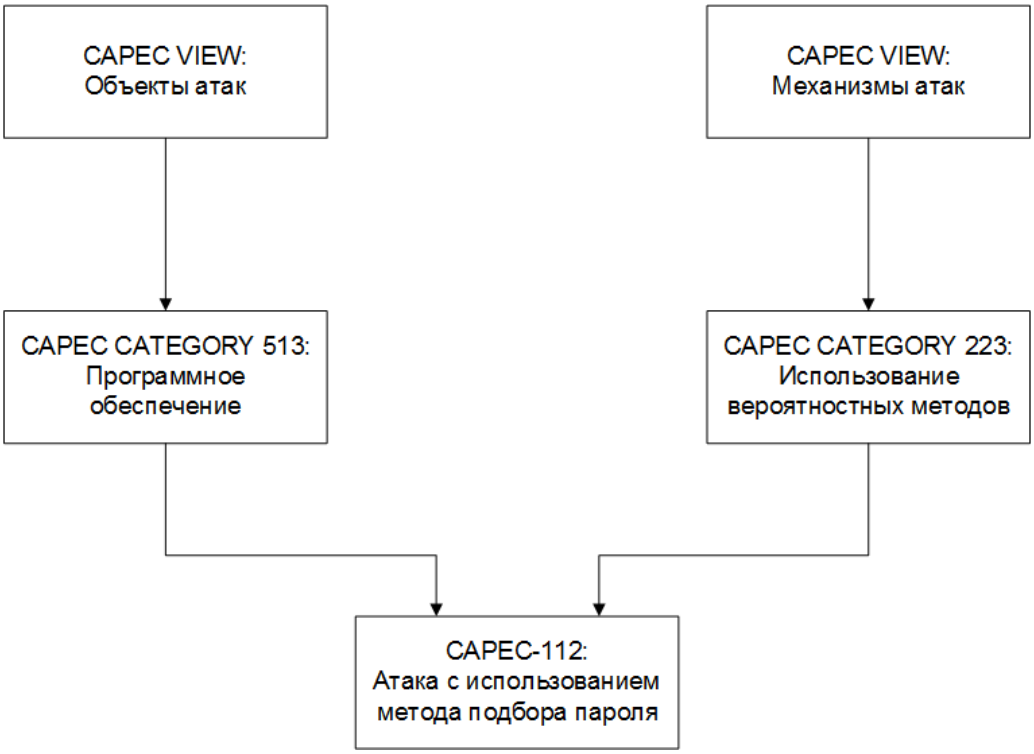

Рисунок 1.3 – Граф связей CAPEC-112 в разных представлениях

Запись о шаблоне атак в CAPEC включает описание механизмов эксплуатации уязвимости, возможные последствия и методы противодействия. CAPEC является иерархической классификацией. Она предоставляют формальный способ описания явлений информационной безопасности, чтобы служить общепринятым языком. Главный упор сделан на их широкое применение, и поэтому MITRE привлекает к разработке специалистов из академического и промышленного сообщества. Эти проекты играют важную роль в обеспечении конструктивного взаимодействия в области информационной безопасности.

Сетевые атаки столь же многообразны, как и системы, против которых они направлены. Некоторые атаки отличаются большой сложностью, другие по силам обычному оператору, даже не предполагающему, к каким последствиям может привести его деятельность. В данной работе для нас представляют интерес атаки, которые могут быть реализованы с помощью HTTP-запросов. Кратко рассмотрим виды таких атак по системе классификации CAPEC и способы борьбы с ними.

272 - Protocol Manipulation

Протокольное манипулирование (Protocol Manipulation) является одним из видов веб-атак, где злоумышленник изменяет или подменяет протокольные данные, передаваемые между клиентом и сервером, с целью выполнения нежелательных действий или получения несанкционированного доступа.

Веб-протоколы, такие как HTTP, TCP/IP и DNS, устанавливают правила и формат обмена данными между клиентом (обычно веб-браузером) и сервером (веб-сервером). Протокольное манипулирование заключается в изменении содержимого или структуры этих данных, чтобы обойти существующие механизмы безопасности или выполнить нежелательные операции.

При HTTP Protocol Manipulation злоумышленник может изменить содержимое HTTP-запросов или ответов, вставлять или изменять параметры, заголовки или тело запроса/ответа. Это может привести к выполнению неправомерных операций на сервере или изменению отображаемого содержимого на стороне клиента.

Несанкционированный пользователь способен перехватывать и изменять TCP/IP-пакеты, передаваемые между клиентом и сервером. Это может включать подделку источника пакета, изменение портов или перенаправление трафика на другие машины.

Злоумышленник может подменить ответы DNS-сервера с целью перенаправления пользователей на нежелательные или фальшивые веб-сайты. Это может привести к фишинговым атакам или перехвату конфиденциальной информации.

Protocol Manipulation представляет серьезную опасность для веб-приложений и пользователей, так как она может привести к различным негативным последствиям, таким как нарушение конфиденциальности данных, подмена и фальсификация данных. нарушение аутентификации и авторизации.

Для борьбы с атаками данного типа рекомендуются следующие меры:

- применение протоколов шифрования, таких как SSL/TLS, что помогает предотвратить перехват и изменение данных во время передачи
- строгая проверка и валидация веб-приложениями входных данных
- применение механизмов контроля целостности, таких как хэширование или цифровые подписи
- мониторинг и регистрация событий для обнаружения аномалий или необычной активности

242 - Code Injection, 88 - OS Command Injection, 66 - SQL Injection

Атаки типа инъекций (Injection) являются распространенным методом атак на веб-приложения и базы данных. Они основаны на внедрении и выполнении вредоносного кода (инъекционных строк) в уязвимые точки веб-приложений, такие как поля ввода или параметры URL-запросов.

Типичные виды инъекций включают SQL-инъекции, код JavaScript-инъекции (XSS), инъекции операционной системы (OSI) и многие другие. В каждом случае целью атакующего является выполнение нежелательных операций или получение несанкционированного доступа к данным или системным ресурсам.

SQL-инъекции являются одним из наиболее распространенных видов атак инъекций. В этом случае злоумышленник вводит вредоносный SQL-код в поля ввода, которые затем передаются базе данных для выполнения. Если веб-приложение недостаточно защищено и не осуществляет достаточной проверки и фильтрации ввода, злоумышленник может выполнить вредоносный SQL-код, получить доступ к данным, изменить или удалить их.

Атака OS Command Injection – это тип веб-атаки, при которой злоумышленник внедряет вредоносные команды операционной системы в веб-приложение, которое впоследствии выполняет эти команды на сервере. Целью атаки является выполнение нежелательных операций на сервере с привилегиями операционной системы. Основная уязвимость, которая позволяет провести атаку OS Command Injection, заключается в некорректной обработке пользовательского ввода, который используется в качестве части командной строки операционной системы без должной проверки или экранирования. Когда злоумышленник вводит специально сформированную строку, содержащую команды операционной системы, веб-приложение выполняет эти команды, что позволяет злоумышленнику выполнить нежелательные действия, такие как создание, изменение или удаление файлов и директорий; управление сетевыми настройками, изменение или удаление записей реестра; запуск исполняемых файлов или скриптов.

Для борьбы с атаками такого типа следует применять следующие методы:

- строгая проверка всех входных данных, поступающих от пользователей
- экранирование любых недопустимых символов или символов, которые могут быть использованы для инъекции кода
- использование параметризованных запросов с предопределенными параметрами

– назначать минимальные привилегии и разрешения для приложений и сервисов, которые выполняют код, особенно для выполнения системных команд

126 - Path Traversal

Атака Path Traversal (также известная как Directory Traversal или Directory Climbing) является типом веб-атаки, при которой злоумышленник пытается получить несанкционированный доступ к файлам и директориям в файловой системе сервера, обойдя ограничения безопасности.

Атака Path Traversal возникает из-за недостаточной проверки и обработки пользовательского ввода, который используется для формирования путей к файлам или директориям на сервере. Злоумышленник может внедрить специально сформированные символы или последовательности символов, которые могут обмануть проверки путей и позволить получить доступ к файлам и директориям, на которые у него нет разрешения.

При добавлении в путь таких символов, как ".." или "../", несанкционированный пользователь может перейти в файловой системе на уровень выше и получить доступ к файлам или директориям, на которые у него нет разрешения. С этой же целью злоумышленники используют URL-кодирование символов. При данном подходе они могут обойти проверки путей и получить доступ к запрещенным файлам или директориям.

Для борьбы с атаками такого типа следует применять следующие методы:

– отклонять любые пути, содержащие запрещенные символы или последовательности символов, включая ".." или "../"
– использовать абсолютные пути к файлам и директориям вместо относительных

– настройка ограничений доступа к файлам и директориям с помощью прав доступа и разрешений файловой системы

16 - Dictionary-based Password Attack

Dictionary-based Password Attack (атака на основе словаря) является методом взлома паролей, при котором злоумышленник использует заранее подготовленный словарь слов или фраз для попыток угадывания пароля. Этот метод основан на предположении, что многие пользователи выбирают слабые или распространенные пароли, которые могут быть найдены в словаре.

Dictionary-based Password Attack происходит следующим образом: злоумышленник создает или получает готовый словарь, содержащий множество слов или фраз, которые могут быть потенциальными паролями. При помощи программы или скрипта происходит автоматический перебор из словаря с целью пройти аутентификацию на целевой системе или сервисе. Злоумышленники могут использовать различные вариации и модификации слов из словаря, например, добавление чисел и символов в конце или в начале строки, замена букв на похожие символы и т.д, для увеличения шансов на успех. Время, необходимое для выполнения атаки на основе словаря, зависит от размера словаря и сложности паролей. Для сильных и уникальных паролей атака на основе словаря может потребовать очень длительного времени или быть неуспешной.

Для борьбы с атаками Dictionary-based Password Attack рекомендуется применять следующие меры:

– установить требования к сложности и уникальности паролей (запрет распространенных паролей и слов, минимальная длина пароля и требования к использованию букв, цифр и символов)
– использование нескольких уровней аутентификации

- механизм автоматической блокировки аккаунтов после нескольких неудачных попыток входа
- ограничение количества попыток входа
- хэширование паролей

310 - Scanning for vulnerable software

Атака Scanning for Vulnerable Software (сканирование уязвимого программного обеспечения) представляет собой процесс поиска и исследования уязвимостей в программном обеспечении, установленном на веб-серверах или других устройствах, связанных с сетью. Злоумышленники сканируют целевые системы, чтобы обнаружить уязвимости, которые могут быть использованы для незаконного доступа, выполнения вредоносных действий или кражи данных.

После сбора информации о целевой системе (IP-адрес, доменных имен или сетевых ресурсов) злоумышленники применяют различные инструменты и скрипты, которые автоматически сканируют целевые узлы и службы на предмет известных уязвимостей. Они ищут уязвимости, такие как открытые порты, слабые пароли, уязвимые версии программного обеспечения или конфигурационные ошибки. После обнаружения уязвимости злоумышленники пытаются эксплуатировать ее для получения несанкционированного доступа или выполнения вредоносных действий. Это может включать использование известных эксплоитов, написание собственного вредоносного кода или запуск специально созданных скриптов и программ.

Методы борьбы с атаками Scanning for Vulnerable Software включают:

- регулярное обновление программного обеспечения, установленного на серверах и сетевых устройствах (обновления часто содержат исправления уязвимостей, которые могут быть использованы злоумышленниками)

- использование специализированных инструментов сканирования уязвимостей для проверки систем на наличие известных уязвимостей
- ведите подробных журналов событий, чтобы отслеживать и анализировать активность в сети и на серверах

153 - Input Data Manipulation

Атака Input Data Manipulation (манипуляция входными данными) является типом веб-атаки, в которой злоумышленник изменяет или подменяет входные данные, передаваемые приложению или сервису через различные формы ввода. Целью такой атаки является обман или нарушение нормального функционирования приложения.

Например, использование другой кодировки символов может привести к тому, что опасный текст будет рассматриваться как безопасный текст. В качестве альтернативы злоумышленник может использовать определенные флаги, такие как расширения файлов, чтобы заставить целевое приложение поверить, что предоставленные данные должны обрабатываться с использованием определенного интерпретатора, когда данные на самом деле не имеют соответствующего типа. Это может привести к обходу механизмов защиты, заставляя цель использовать определенные компоненты для обработки входных данных или, иным образом приводя к обработке пользовательских данных иначе, чем можно было бы ожидать.

274 - HTTP Verb Tampering

Атака «HTTP Verb Tampering» (подделка HTTP-глаголов) - это тип веб-атаки, при которой злоумышленник изменяет или подменяет HTTP-глаголы в запросах, отправляемых клиентом на сервер. Эта атака использует возможность изменить метод запроса HTTP, чтобы обмануть сервер и выполнить нежелательные операции. Злоумышленник изменяет HTTP-глагол (например, GET, PUT, TRACE и т.д.), чтобы обойти ограничения

доступа. Некоторые веб-среды позволяют администраторам ограничивать доступ на основе HTTP-глагола, используемого при запросах. Однако злоумышленники часто могут использовать другой HTTP-глагол или даже использовать случайную строку в качестве глагола, чтобы обойти эти средства защиты. Это позволяет злоумышленнику получить доступ к данным, которые в противном случае должны быть защищены.

Если сервер основывается на методе POST для обработки конкретного действия (например, удаление или изменение данных), злоумышленник может изменить метод на GET и получить доступ к операции без необходимой аутентификации или авторизации.

Если сервер неправильно проверяет методы PUT или DELETE и позволяет выполнить операции изменения или удаления данных с использованием этих методов, злоумышленник может изменить метод на GET и получить доступ к операциям изменения или удаления без должной проверки.

Методы борьбы с атаками Scanning for Vulnerable Software включают:

- должна осуществляться проверка и фильтрация входных данных, включая методы запроса HTTP (метод запроса должен соответствовать ожидаемому типу операции)
- проверка аутентификации и авторизации
- использование CSRF-токенов (маркеров) может помочь предотвратить атаки, связанные с изменением HTTP-глаголов (CSRF-токен должен быть включен в каждый запрос и проверяться на сервере для подтверждения подлинности запроса)
- правильная настройка серверов таким образом, чтобы они корректно обрабатывали и проверяли методы запроса HTTP, основываясь на некорректных или измененных методах

194 - Fake of source data

Атака Fake of source data (подделка исходных данных) - это тип веб-атаки, при которой злоумышленник подменяет или подделывает исходные данные, поступающие от источника, перед их обработкой или использованием в приложении или сервисе.

Злоумышленник использует в своих интересах неправильную аутентификацию для предоставления данных или услуг под фальсифицированным удостоверением личности. Целью использования фальсифицированной идентификационной информации может быть предотвращение отслеживания предоставленных данных или присвоение прав, предоставленных другому лицу. Одной из простейших форм этой атаки было бы создание сообщения электронной почты с измененным полем «От», чтобы создать впечатление, что сообщение было отправлено от кого-то другого, а не от фактического отправителя. Источник атаки (в данном случае система электронной почты) не может должным образом аутентифицировать источник, и это приводит к неправильному выполнению читателем предписанного действия. Результаты атаки различаются в зависимости от деталей атаки, но общие результаты включают повышение привилегий, обфускацию других атак и повреждение / манипулирование данными.

Атака может нарушить целостность данных, так как подделанные или измененные данные могут привести к неправильной обработке или хранению информации. Это может иметь серьезные последствия для приложений и систем, которые полагаются на точность и целостность данных. Вполне возможны финансовые потери, особенно если искажение данных влияет на финансовые расчеты, транзакции или договорные обязательства. Это может привести к неправильным расчетам, неправильным платежам или другим финансовым негативным последствиям.

34 - HTTP Response Splitting

HTTP Response Splitting (разделение HTTP-ответа) - это веб-атака, которая позволяет злоумышленнику внедрить дополнительный HTTP-ответ в ответ, отправляемый сервером на запрос клиента. Атака основывается на использовании недопустимых символов, таких как перевод строки или перенос строки, для разделения заголовков HTTP-ответа. Целью атаки является введение дополнительного содержимого, которое может привести к различным последствиям, включая XSS (межсайтовый скриптинг), перенаправление пользователя на злоумышленный сайт или обнаружение информации.

При атаке HTTP Response Splitting, злоумышленник вводит вредоносные данные, которые впоследствии используются в различных стандартных и/или пользовательских HTTP-заголовках HTTP-ответа. Эти данные содержат символы, такие как возврат каретки (CR), перевод строки (LF), горизонтальная табуляция (HT), пробел (SP) и другие символы, соответствующие спецификации RFC, а также специальные символы и уникальные кодировки символов.

В конечном итоге один и тот же HTTP-ответ разбивается на два или более HTTP-ответа при обработке клиентским HTTP-агентом. Злоумышленник производит манипуляции с HTTP-ответом, чтобы обойти механизмы безопасности и доставить вредоносный контент. Это может привести к компрометации приложений и пользователей, а также предоставлению доступа к конфиденциальным данным. Для достижения этой цели злоумышленник эксплуатирует недочеты в синтаксическом анализе и обработке HTTP-ответов на промежуточных HTTP-агентах (например, балансировщики нагрузки, обратные прокси-серверы, прокси-серверы веб-кэширования, брандмауэры приложений и др.) или клиентских HTTP-агентах (например, веб-браузерах).

Эта атака обычно происходит из-за использования устаревших или несовместимых версий протокола HTTP, а также недостаточной проверки

синтаксиса и фильтрации пользовательского ввода в HTTP-агентах, которые принимают HTTP-сообщения в процессе передачи по сети.

33 - HTTP Request Smuggling

Атака HTTP Request Smuggling (смешение запросов HTTP) является сложной и опасной уязвимостью, которая может быть использована злоумышленниками для обхода средств безопасности и осуществления вредоносных действий веб-приложения. Эта атака включает в себя манипуляцию HTTP-запросами и эксплуатацию несоответствий в различных компонентах, обрабатывающих эти запросы.

Злоумышленник эксплуатирует гибкость и несоответствия при обработке и интерпретации HTTP-запросов, используя различные HTTP-заголовки, параметры строки запроса и тело сообщений, а также размеры сообщений (определяемые определенными HTTP-заголовками), в различных промежуточных HTTP-агентах (например, балансировщики нагрузки, обратные прокси-серверы, прокси-серверы кэширования, брандмауэры приложений и другие). Целью является скрытная отправка несанкционированных и вредоносных HTTP-запросов внутреннему HTTP-агенту (например, веб-серверу).

Вредоносный HTTP-запрос, содержащий скрытый второй HTTP-запрос, обрабатывается промежуточным веб-прокси-сервером как один безопасный запрос и перенаправляется на внутренний сервер. Внутренний сервер интерпретирует и анализирует этот запрос как два допустимых безопасных запроса, обходя средства контроля безопасности.

Атака обычно использует некорректное использование HTTP-заголовков, таких как Content-Length и Transfer-Encoding. Кроме того, атака может включать модификацию и искажение параметров строки запроса HTTP-сообщений. Эта атака часто связана с использованием устаревших или несовместимых версий протокола HTTP в HTTP-агентах.

1.6. Выводы по главе

В данной главе была обозначена актуальность задачи выявления веб-атак на основе HTTP-запросов, были рассмотрены наиболее распространенные средства защиты от веб-атак: мониторинг сетевого трафика, системы обнаружения веб-атак (WAF) , RASP (Runtime Application Self-Protection), системы мониторинга событий безопасности (SIEM), средства обнаружения компьютерных атак на конечных устройствах (EDR), системы анализа сетевого трафика (NTA), система анализа сетевого трафика нового поколения (NDR), поиск угроз (Threat hunting). Все эти средства предлагают различные подходы и компетенции к обнаружению веб-атак.

Защита от атак, основанных на HTTP-запросах, остается актуальной и критически важной в современной сетевой среде. Поскольку протокол HTTP широко используется веб-приложениями и сервисами, HTTP-запросы являются основным каналом передачи данных и коммуникации между клиентами и серверами. В связи с этим, атакующие могут использовать HTTP-запросы для эксплуатации уязвимостей и осуществления различных видов атак, поэтому необходима соответствующая защита.

Также была изучена система классификации CAPEC, и разобраны основные типы атак, которые в дальнейшем будут выявляться в данной работе.

Глава 2. ПОДХОДЫ К ВЫЯВЛЕНИЮ ВЕБ-АТАК С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ

2.1. Задача классификации

Постановка задачи классификации в машинном обучении связана с определением класса или категории, к которой принадлежит некоторый объект или наблюдение на основе набора признаков. Это одна из основных задач в области обучения с учителем. Формальная постановка задачи: отнесение исходных данных к определенному классу путем выявления отличительных признаков, характеризующих это данные, в отличии от общей массы данных. Математическая постановка задачи: пусть X – множество описаний объектов, Y – множество номеров наименований классов. Существует неизвестная целевая зависимость – отображение $y^*: X \to Y$, значения которой известны только на объектах конечной обучающей выборки $X^m = \{(x_1, y_1), \dots, (x_m, y_m)\}$. Требуется построить алгоритм $\alpha: X \to Y$, способный классифицировать произвольный объект x, принадлежащий X.

В машинном обучении существует несколько видов классификации, которые отличаются по своим особенностям и подходам. Вот некоторые из них:

- Бинарная классификация: это тип классификации, где объекты разделяются на два класса. Например, задача определения, является ли электронное письмо спамом или не-спамом, является бинарной классификацией, где два класса – «спам» и «не-спам».
- Многоклассовая классификация: в этом случае объекты классифицируются на более чем два класса. Например, распознавание рукописных цифр от 0 до 9 является задачей многоклассовой классификации, где каждая цифра представляет отдельный класс.

– Multi-label классификация: этот вид классификации аналогичен многоклассовой классификации, но с различными особенностями. В Multi-label классификации каждый объект может принадлежать одновременно к нескольким классам.
– Одноклассовая классификация: в этом случае задача состоит в том, чтобы классифицировать объекты только на основе их сходства с одним классом, не требуя информации о других классах. Одноклассовая классификация обычно используется для задач обнаружения аномалий или выбросов, где цель состоит в выявлении объектов, которые не соответствуют нормальному или ожидаемому поведению.

Классификация веб-атак с использованием методов машинного обучения является эффективным подходом для автоматизации процесса обнаружения и предотвращения атак. Задача классификации заключается в обучении модели на основе исторических данных об атаках и их признаках, а затем использовании этой модели для определения новых атак на основе их характеристик.

Ниже приведен общий процесс классификации веб-атак с использованием машинного обучения:

1. **Подготовка данных**: сначала необходимо собрать и подготовить набор данных, который будет использоваться для обучения модели. Этот набор данных должен содержать информацию о различных типах веб-атак и соответствующих им признаках или характеристиках.
2. **Извлечение признаков:** для обучения модели необходимо извлечь значимые признаки из сырых данных. Это может включать такие характеристики, как IP-адрес отправителя, тип запроса, заголовки HTTP, параметры URL и другие свойства запросов.
3. **Выбор и обучение модели:** следующий шаг - выбор подходящей модели машинного обучения, которая будет использоваться для классификации веб-атак. Некоторые из популярных моделей включают

логистическую регрессию, метод опорных векторов (SVM), случайные леса, градиентный бустинг и нейронные сети. Модель обучается на подготовленных данных, чтобы научиться выявлять и различать различные типы атак.

4. **Оценка и настройка модели:** после обучения модели ее необходимо оценить с использованием тестового набора данных, который не использовался в процессе обучения. Это позволяет оценить точность и эффективность модели. Если необходимо, модель может быть доработана и настроена, чтобы достичь лучших результатов.
5. **Применение модели:** после успешного обучения и оценки модель можно использовать для классификации новых веб-атак. На основе характеристик атаки модель определит ее тип и примет соответствующие меры для предотвращения или ограничения воздействия атаки.

Важно отметить, что эффективность обнаружения веб-атак с помощью машинного обучения зависит от нескольких факторов: качество и разнообразие данных (набор данных должен содержать достаточное количество примеров различных типов атак, чтобы модель могла научиться распознавать их; данные должны быть достаточно представительными для реальных условий и сценариев, чтобы модель могла обобщать на новые атаки), выбор признаков (выбор релевантных и информативных признаков влияет на способность модели распознавать атаки; хорошо выбранные признаки должны быть связаны с характеристиками атаки и в то же время отличаться от нормального поведения, чтобы модель могла их различать; процесс извлечения признаков требует хорошего понимания домена и атак, чтобы правильно определить релевантные характеристики), а также выбор подходящей модели обучения и настройка ее параметров.

В данной работе для задачи классификации веб-атак было принято решение воспользоваться следующими моделями обучения: Random Forest и

XGBosst. Ниже приведено подробное описание и принцип работы каждой из этих моделей.

Random Forest (случайный лес)

RandomForest - это алгоритм машинного обучения, который используется для задач классификации, регрессии и других типов анализа данных. Он основан на идее ансамблей деревьев решений и комбинирует прогнозы нескольких деревьев для достижения более точного и стабильного предсказания.

Основной строительный блок модели RandomForest - это деревья решений. Дерево решений - это иерархическая структура, состоящая из узлов и ребер, где каждый узел представляет тест на определенное свойство данных, а каждое ребро представляет возможный результат этого теста. Деревья решений разбивают данные на основе различных признаков, чтобы сформировать прогноз или классификацию.

RandomForest строит ансамбль деревьев решений. Это означает, что он создает множество деревьев, где каждое дерево обучается на подмножестве данных и с использованием подмножества признаков. Каждое дерево делает независимый прогноз, и итоговый прогноз RandomForest определяется голосованием или усреднением прогнозов всех деревьев.

Одним из первых и самых простых видов ансамблей считается Bagging (от Bootstrap aggregation), который и лежит в основе алгоритма случайного леса.

Bagging - это метод ансамблевого обучения в машинном обучении, который заключается в комбинировании прогнозов нескольких базовых моделей для улучшения обобщающей способности и стабильности предсказаний. Бэггинг использует метод bootstrap для генерации случайных подвыборок из исходного обучающего набора данных. Это означает, что каждая подвыборка формируется путем выбора объектов из обучающего набора с повторениями. Размер каждой подвыборки обычно равен исходному

набору данных, но это необязательное условие. На рисунке 2.1 схематично представлена работа этого метода.

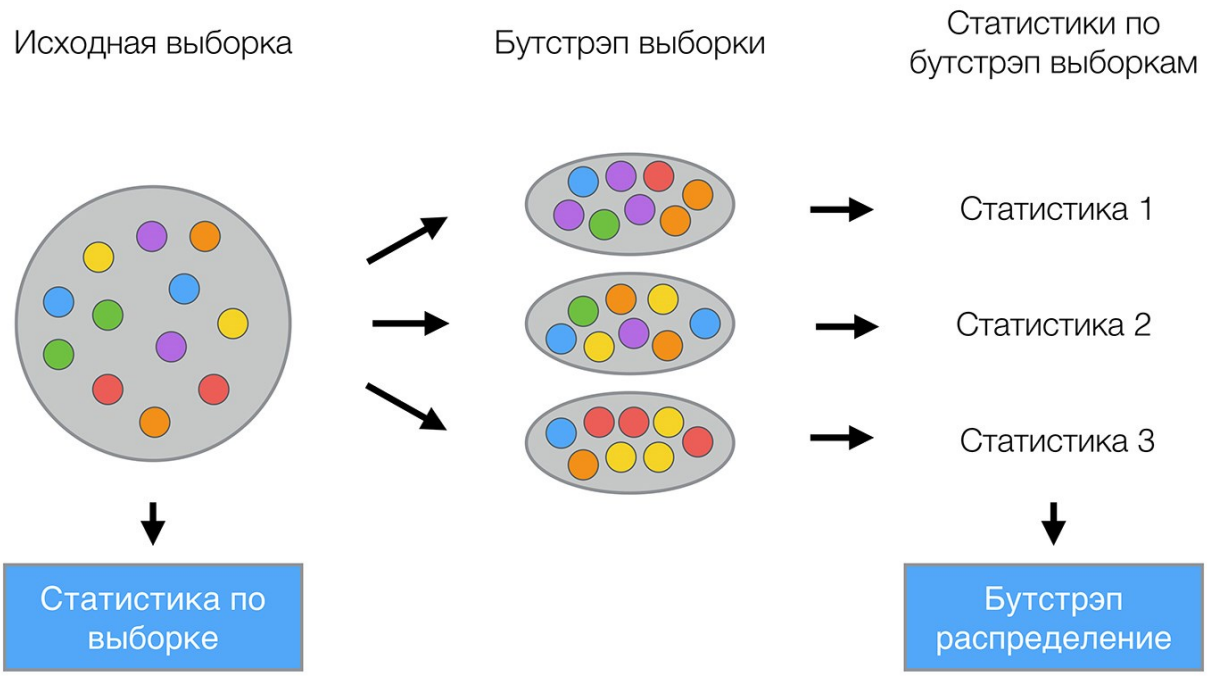

Рисунок 2.1 – Схематичное представление работы метода bootstrap

После обучения всех базовых моделей, бэггинг комбинирует их прогнозы для получения итогового предсказания. В задачах классификации часто используется голосование большинства, где класс, выбранный большинством моделей, считается окончательным предсказанием. В задачах регрессии используется усреднение прогнозов базовых моделей. Этот процесс представлен на рисунке 2.2.

Random Forest обладает высокой точностью предсказания в широком спектре задач классификации и регрессии. Он может обрабатывать сложные и нелинейные зависимости в данных, делая его мощным инструментом для прогнозирования. Random Forest способен обрабатывать большие наборы данных с множеством признаков, не подвергаясь переобучению. Это связано с использованием бэггинга и случайного выбора признаков для каждого дерева, что способствует разнообразию и независимости деревьев.

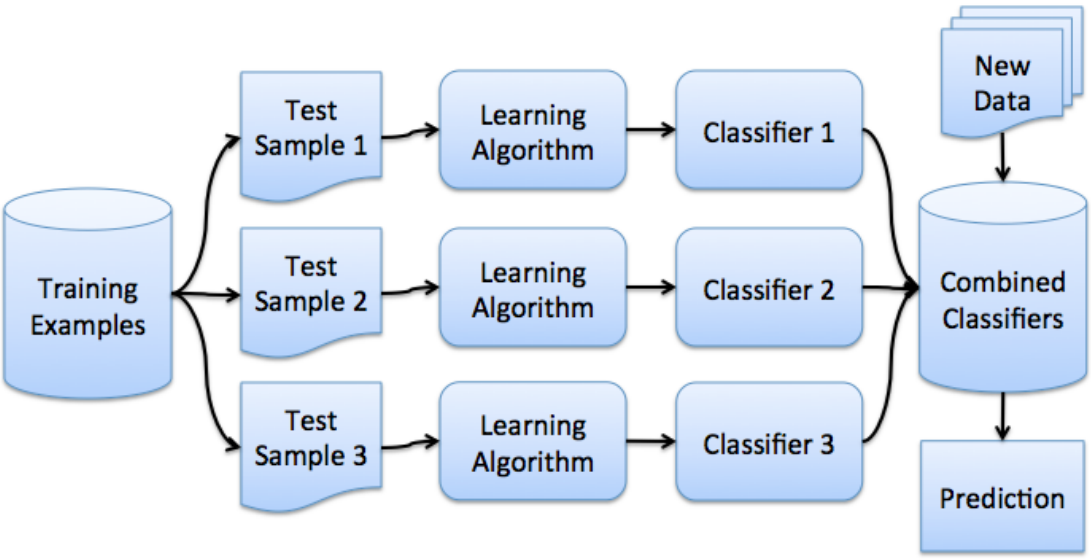

Рисунок 2.2 – Схематичное представление алгоритма Bagging

Random Forest может работать с различными типами данных, включая категориальные и числовые признаки, без необходимости предварительной обработки или масштабирования. Алгоритм предоставляет информацию о важности каждого признака, что помогает понять, какие признаки наиболее сильно влияют на предсказания модели. Это может быть полезно для отбора признаков и понимания данных.

Random Forest обычно хорошо справляется с выбросами в данных благодаря среднему значению предсказаний из множества деревьев, что делает его устойчивым к шуму и аномалиям.

XGBoost (eXtreme Gradient Boosting)

В процессе эволюции алгоритмов, основанных на дереве поиска решений, появился новый метод ансамблирования решающих деревьев под названием Бустинг (boosting). Бустинг (boosting) - это алгоритм машинного обучения, который комбинирует несколько слабых моделей обучения (называемых базовыми моделями или учащимися) для создания более сильной и предсказательной модели. Он относится к семейству ансамблевых методов, где модели объединяются для улучшения общей

производительности. Бустинг чем-то похож на бэггинг, но в бэггинге модели обучаются совершенно независимо и параллельно, а в бустинге последовательно, с оглядкой на предыдущие.

Все началось с исследования возможности объединения большого числа относительно слабых и простых моделей в одну сильную модель. Здесь под слабыми моделями подразумеваются модели, которые имеют ограниченную точность и сложность, в отличие от более «сильных» моделей, таких как нейронные сети. В данном случае слабые модели представляют собой произвольные алгоритмы машинного обучения, чья точность может быть незначительно выше случайного угадывания.

Математический ответ на этот вопрос был положительным и был получен довольно быстро, что само по себе является значимым теоретическим результатом в области машинного обучения. Однако, прошло несколько лет до того, как были разработаны работоспособные алгоритмы, такие как Adaboost. Основной подход состоял в пошаговом построении линейной комбинации простых моделей (базовых алгоритмов) путем взвешивания входных данных. Каждая последующая модель, обычно дерево решений, строилась таким образом, чтобы уделять больший вес и приоритет ранее неправильно предсказанным наблюдениям.

XGBoost (eXtreme Gradient Boosting) - это мощный алгоритм машинного обучения, основанный на градиентном бустинге деревьев решений. Он широко используется в различных задачах, включая классификацию, регрессию и ранжирование. XGBoost представляет собой улучшенную версию градиентного бустинга, обладающую высокой эффективностью и точностью предсказания.

Вместо применения традиционного подхода, где обучается одна модель на основе всех данных, применяется метод, который включает обучение множества моделей на разных подмножествах обучающей выборки. После этого проводится голосование и выбирается модель с лучшими результатами.

XGBoost использует градиентный спуск для оптимизации функции потерь модели. Он итеративно обучает деревья решений, минимизируя градиент функции потерь, что позволяет модели постепенно улучшаться.

XGBoost оптимизирован для эффективной работы на больших наборах данных. Он использует параллельные вычисления и оптимизированные структуры данных для ускорения процесса обучения и предсказания.

Данный алгоритм включает различные методы регуляризации для предотвращения переобучения модели. Некоторые из них включают ограничение глубины деревьев, регуляризацию весов и функции потерь, а также использование случайной выборки объектов и признаков.

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}^{(t-1)} + f_t(x_i)) + \Omega(f_t), \quad (1)$$

где $L^{(t)}$ - функция для оптимизации градиентного бустинга,

l - функция потерь,

$y_i, \hat{y}^{(t)}$ - значение i-го элемента обучающей выборки и сумма предсказаний первых t деревьев соответственно,

x_i - набор признаков i-го элемента обучающей выборки,

f_t - функция, которая обучается на шаге t,

$f_t(x_i)$ - предсказание на i-ом элементе обучающей выборки,

$\Omega(f)$ - регуляризация функции f.

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \| \omega \|^2, \quad (2)$$

где γ, λ - параметры регуляризации,

T - количество вершин в дереве,

ω - значение в листьях.

Поскольку мы хотим минимизировать ошибку модели на обучающей выборке, нам нужно найти минимум $L^{(t)}$ для каждого t. Каждое отдельное дерево ансамбля $f_t(x_i)$ обучается стандартным алгоритмом.

Для оценки качества классификации было принято решение использовать следующие показатели: Accuracy, Precision, Recall, F1-score.

Accuracy: показывает долю объектов, для которых класс был предсказан верно; данная метрика характеризует качество модели, агрегированное по всем классам.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad (3)$$

где TP (True Positive) – доля верно классифицированных положительных примеров,

TN (True Negative) – доля верно классифицированных отрицательных примеров,

FP (False Positive) – доля неверно классифицированных положительных примеров (ошибка I рода),

FN (False Negative) – доля неверно классифицированных отрицательных примеров (ошибка II рода).

Precision: показывает долю правильно предсказанных положительных объектов среди всех объектов, предсказанных положительным классом.

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

Recall: показывает долю правильно найденных положительных объектов среди всех объектов положительного класса.

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

F1-score: среднее гармоническое между Precision и Recall; данная метрика предполагает одинаковую важность между ними.

$$F1 - score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (6)$$

2.2. Способы предварительной обработки текстовых данных

Перед обучением различных моделей машинного обучения необходим обзор и предварительная обработка данных. Основная цель предварительной обработки данных и выбора соответствующих функций набора данных состоит в том, чтобы позволить различным алгоритмам и моделям машинного обучения достичь максимального уровня точности в своих прогнозах. Предварительная обработка данных при работе с веб-запросами имеет свои особенности, связанные со спецификой данных, которые получаются из веб-среды. Они имеют схожие аспекты с текстовыми данными, такие как наличие текстовой информации, но существуют определенные особенности, требующие своей специфики работы.

Предобработка текстовых данных обычно включает в себя следующие основные этапы:

- разбиение текста на отдельные слова
- отсечение стоп-слов (стоп-слова – слова, не несущие смысловой информации, например, местоимения, предлоги, союзы, артикли и т.д.)
- стемминг (процесс нахождения основы слова для заданного исходного слова; основа необязательно совпадает с морфологическим корнем слова)
- лемматизация (процесс приведения слова к его нормальной форме; нормальной формой слова является форма единственного числа в именительном падеже для русского языка, и форма единственного числа в инфинитиве для английского языка)
- взвешивание терминов (определение их меры информативности)

Чаще всего для взвешивания используют tf - взвешивание (term-frequency):

$$x_j^{(i)} = f_{ij}, \tag{7}$$

где f_{ij} - частота появления термина i в документе j, а также $tf - idf$ - взвешивание [10], в котором вес слова i в документе j пропорционален числу вхождений слова в данный документ и обратно пропорционален общему числу документов в выборке, в которых также содержится это слово:

$$x_j^{(i)} = f_{ij} \log\left(\frac{N}{N_i}\right), \tag{8}$$

где f_{ij} - частота появления термина i в документе j,

N_i - число документов выборки, в которых встретилось слово i,

N - общее число документов выборки.

Однако веб-запросы могут содержать не только текстовую информацию, но и другие типы данных, такие как числовые значения, специальные символы, временные метки и т.д. При этом для идентификации угрозы необходима полная информация о содержании и структуре веб-запроса, поэтому такие методы, как отсечение стоп-слов, лемматизация, стемминг не подходят.

Веб-запросы могут содержать специфические признаки, такие как IP-адреса, даты и временные метки, информацию о сеансе пользователя и другие метаданные. Эти признаки требуют специальной обработки и преобразования для использования в классификации. Также не стоит забывать и о сетевых аспектах (порт, протокол), URL-кодировках, содержащих специфические символы, кодировку или параметры запроса.

Особое внимание следует уделить отбору информативных признаков. Веб-запросы могут содержать, например, HTML-теги и другую разметку, которая не несет смысловой нагрузки для классификации текста. Эта разметка должна быть удалена или обработана перед анализом текста.

Все эти особенности не позволяют пользоваться привычными для нас методами предобработки данных. Целью данной работы является разработка

и проверка методов численной предобработки данных веб-запросов для последующей их классификации с целью выявления веб-атак.

2.3. Выводы по главе

В данной главе была приведена постановка задачи классификации веб-атак, приведены общие этапы классификации веб-атак с помощью машинного обучения. Была обозначена важность и особенности предварительной обработки данных, основанных на веб-запросах.

Также были описаны модели классификаторов, которые будут использоваться в данной работе – Random Forest и XGBoost (eXtreme Gradient Boosting). Ансамблевые модели, такие как случайный лес или градиентный бустинг, могут повысить точность и надежность обнаружения веб-атак. Комбинирование прогнозов нескольких моделей может снизить вероятность ложных срабатываний и улучшить общее качество классификации.

Глава 3. ПРОВЕДЕНИЕ ИССЛЕДОВАНИЙ ПО ВЫЯВЛЕНИЮ И КЛАССИФИКАЦИИ ВЕБ-АТАК НА ОСНОВЕ HTTP-ЗАПРОСОВ.

3.1. Описание выборки.

Для экспериментов и оценки различных моделей в данной работе был использован набор данных SR-BH 2020. Набор данных состоит из веб-запросов, собранных веб-сервером Wordpress. На этом сервере была установлена версия 2.9.2 Modsecurity в режиме «Только обнаружение», позволяющая регистрировать все запросы (вредоносные и нормальные) в журнале, сгенерированном Modsecurity, но без блокировки.

По истечении времени воздействия на веб-сервер, собранные журналы были анализированы авторами с использованием как ручных, так и полуавтоматических методов для проверки разметки веб-запросов. При необходимости, метки запросов были исправлены. Каждому веб-запросу была присвоена метка «обычный» или «атака» в соответствии с классификацией CAPEC, чтобы обеспечить соответствие и согласованность с данными классификации.

Результатом является специально созданный набор данных с несколькими метками, который предназначен для обнаружения веб-атак. Набор данных состоит из 907 814 веб-запросов, из которых 525 195 являются обычными, а 382 619 - аномальными. Каждая запись содержит 24 различных признака и 13 меток. Первая метка указывает, является ли веб-запрос нормальным (1) или нет (0). Если значение первой метки равно 1 (обычный запрос), то остальные метки в наборе данных должны быть равны 0. В противном случае, если значение первой метки равно 0 (возможная атака), в наборе данных должна присутствовать одна или несколько меток со значением 1.

Данные о количестве веб-запросов, классифицируемых по различным идентификаторам CAPEC, представлены в подробной форме в таблице 1. Важно отметить следующее: общее количество веб-запросов в таблице 1

превышает количество запросов в наборе данных. Это объясняется тем, что некоторые веб-запросы сочетают в себе более одного типа атаки. Дополнительная информация о количестве веб-запросов с несколькими метками представлена в таблице 2.

Таблица 1 – Количество веб-запросов по классификации CAPEC

Классификация CAPEC	**Количество веб-запросов**	**% от общего числа запросов**
000 - Normal	525195	57.85%
272 - Protocol Manipulation	9153	1.00%
242 - Code Injection	15827	1.74%
88 - OS Command Injection	7482	0.82%
126 - Path Traversal	20992	2.31%
66 - SQL Injection	250311	27.57%
16 - Dictionary-based Password Attack	1847	0.20%
310 - Scanning for vulnerable software	2718	0.30%
153 - Input Data Manipulation	2272	0.25%
274 - HTTP Verb Tampering	5437	0.60%
194 - Fake the source of data	56145	6.18%
34 - HTTP Response Splitting	19738	2.17%
33 - HTTP Request Smuggling	1059	0.12%
ВСЕГО	918176	

Таблица 2 – Количество различных классификаций CAPEC, назначенных веб-запросу

Количество разных классификаций CAPEC	**Количество веб-запросов**
1	898576
2	8132
3	1088
4	18

Набор данных SR-BH 2020 является первым в своем роде, предоставляющим возможность обучения и оценки моделей и алгоритмов машинного обучения с использованием множественных меток, которые

позволяют классифицировать атаки CAPEC, направленные на веб-приложения [11].

Набор данных содержит следующие поля:

- дата и время веб-запроса (timestamp)
- IP-адреса получателя и отправителя (src_ip, dst_ip)
- порт получателя и отправителя (src_port, dst_port)
- метод запроса (request_http_method),
- протокол запроса (request_http_protocol)
- используемый агент запроса (request_user_agent)
- адрес хоста (request_host)
- тип соединения (request_connection)
- тело запроса (request_body)
- язык и кодировка запроса (request_accept_language, request_accept_encoding)

3.2. Применение различных подходов к обработке запросов

3.2.1. Вычисление средней суммы ASCII символов

Данный подход к предварительной обработке данных основан на вычислении средней суммы значений ASCII символов для каждого поля веб-запроса. При этом символы преобразуются в нижний регистр. Таким образом, поля, в которых содержится большое количество аномальных символов, таких как "./../." в типичных попытках атаки на обход пути, будут иметь отличающиеся значения ASCII по сравнению с полями, где веб-запросы являются нормальными. Подробная процедура примера предлагаемого числового кодирования представлена на рисунке 3.1.

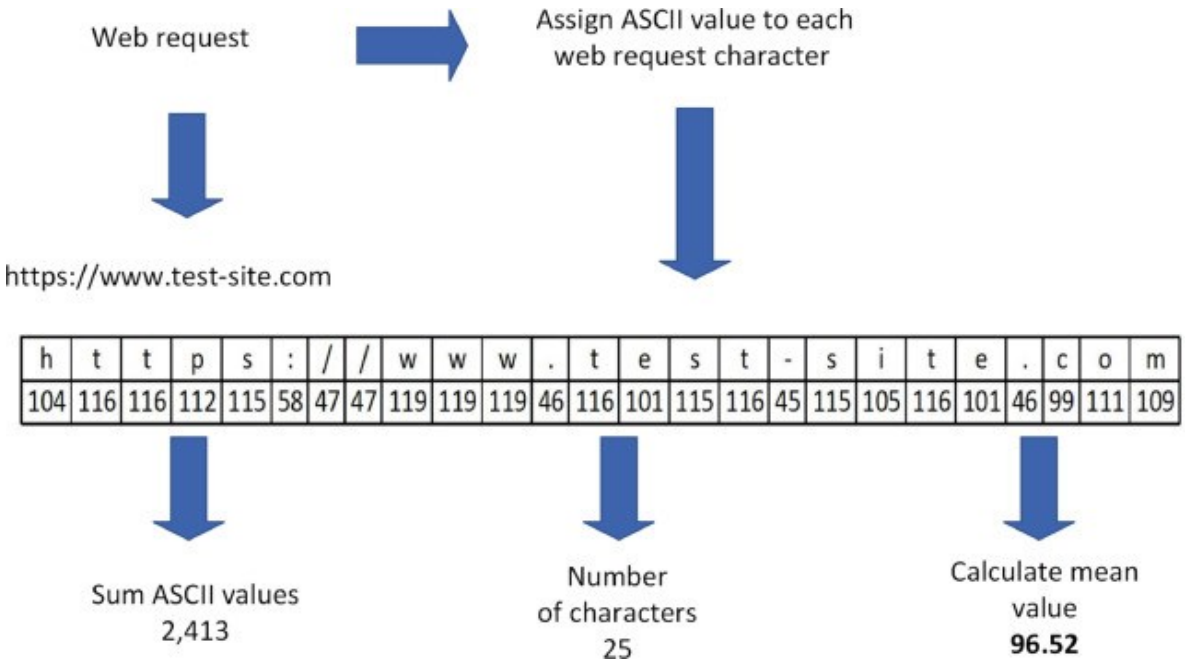

Рисунок 3.1 – Пример численного кодирования веб-запроса

На данном этапе проверялась следующая гипотеза: среднее значение суммы ASCII символов для атакующего запроса будут отличными по сравнению с нормальным из-за наличия, например, большого количества аномальных символов. Таким образом, с помощью этой разницы классификатор и будет отличать нормальный запрос от атакующего.

Для примера было проведено сравнение значений различных полей обычного и атакующего веб-запроса. В таблице 3 приведена информация о

разнице значений средних значений суммы ASCII символов. В таблице 4 представлена подробная информация о полях веб-запроса, в которых наблюдаются отличия в средней сумме их значений ASCII.

Таблица 3 – Средние значения полей ASCII обычного веб-запроса и атаки

Поле	**Среднее значение обычного запроса**	**Среднее значение запроса-атаки**	**Разница**
method_value	106.667	113.5	ДА
http_request_value	97.1	94.7	ДА
protocol_value	79.875	79.875	НЕТ
referer_value	105.667	105.667	НЕТ
agent_value	73.9618	98.4444	ДА
host_value	99.6154	48.5556	ДА
origin_value	105.667	105.667	НЕТ
cookie_value	105.667	105.667	НЕТ
content_type_value	105.667	100.848	ДА
accept_value	43.6667	43.6667	НЕТ
accept_language_value	105.667	105.667	НЕТ
accept_encoding_value	95.6154	95.6154	НЕТ
do_not_track_value	0	0	НЕТ
connection_value	99.5	99.5	НЕТ
body_value	105.667	92.7102	ДА
http_status_code_value	200	404	ДА
http_status_message_value	109	101	ДА

Таблица 4 – Подробная информация о полях с различными средними значениями ASCII

Поле	**Обычный веб-запрос**	**Атака**
method_value	GET	POST
http_request_value	/blog/xmlrpc.php?rsd	/cgi-bin/ViewLog.asp

Продолжение таблицы 4

agent_value	Mozilla/4.0 (compatible;...)	B4ckdoor-owned-you
host_value	test-site.com	127.0.0.1
content_type_value	nan	application/x-www-form-urlencoded
body_value	nan	remote_submit_Flag=1& remote_syslog_Flag=1& RemoteSyslogSupported=1&LogFlag=0 & remote_host=%3bcd+/tmp;wget+ http://45.95.168.230/ taevimncorufglbzhwxqpdkjs/Meth.arm7 ;chmod+777+Meth
http_status_code_value	200	404
http_status_message_value	OK	404
response_content_length_value	317	271

После численного преобразования каждого поля веб-запроса необходимо очистить выборку от признаков, которые не предоставляют полезной информации (неинформативные признаки). На рисунке 3.2 представлена информативность каждой функции поля веб-запроса.

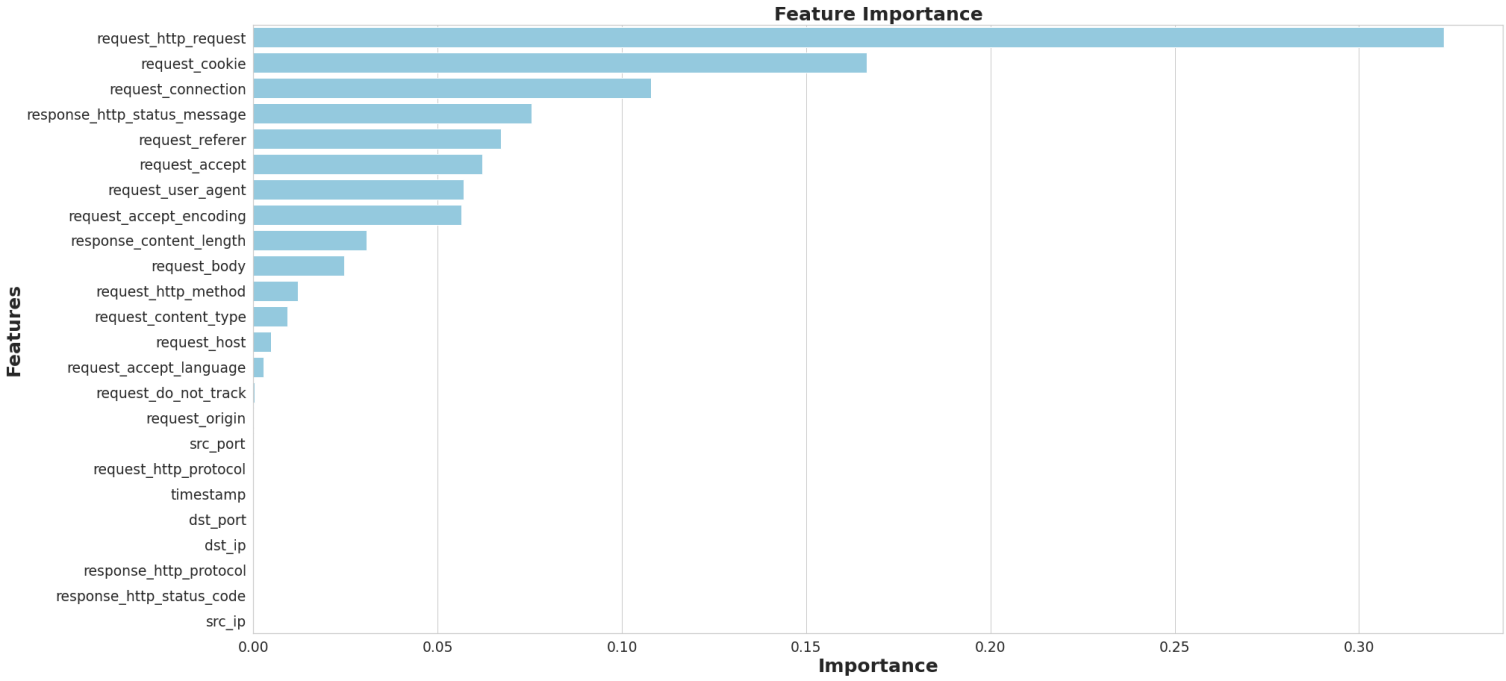

Рисунок 3.2 – Информативность признаков поля веб-запроса

Как только признаки, которые не представляют полезной информации, были удалены, оставшиеся признаки стандартизируются с использованием класса StandardScaler библиотеки Scikit-learn.

Наконец, после предварительной обработки и отбора информативных признаков, были обучены 2 модели классификации: Random Forest (случайный лес) и XGBoost.

В таблице 5 и таблице 6 приведены результаты классификации для выбранных показателей качества для 2 моделей: Random Forest и XGBoost соответственно.

Таблица 5 – Результаты классификации для вычисления суммы ASCII символов для модели Random Forest

	Precision	**Recall**	**F1-score**
000 - Normal	0.94	0.92	0.93
272 - Protocol Manipulation	0.93	0.52	0.66
242 - Code Injection	0.94	0.70	0.80
88 - OS Command Injection	0.95	0.35	0.51
126 - Path Traversal	0.91	0.76	0.83
66 - SQL Injection	0.87	0.85	0.86

Продолжение таблицы 5

16 - Dictionary-based Password Attack	1.00	0.86	0.92
310 - Scanning for Vulnerable Software	1.00	1.00	1.00
153 - Input Data Manipulation	0.16	0.97	0.28
274 - HTTP Verb Tampering	1.00	0.99	0.99
194 - Fake the Source of Data	0.80	0.42	0.55
34 - HTTP Response Splitting	0.63	0.37	0.46
33 - HTTP Request Smuggling	0.95	0.73	0.82
weighted avg	0.91	0.84	0.87
Accuracy	0.84263980		

Таблица 6 – Результаты классификации для вычисления суммы ASCII символов для модели XGBoost

	Precision	**Recall**	**F1-score**
000 - Normal	0.93	0.83	0.88
272 - Protocol Manipulation	0.96	0.24	0.38
242 - Code Injection	0.88	0.31	0.45
88 - OS Command Injection	0.96	0.01	0.02
126 - Path Traversal	0.83	0.32	0.46
66 - SQL Injection	0.87	0.70	0.77
16 - Dictionary-based Password Attack	0.98	0.42	0.59
310 - Scanning for Vulnerable Software	1.00	0.90	0.95
153 - Input Data Manipulation	0.10	0.97	0.19
274 - HTTP Verb Tampering	1.00	1.00	1.00
194 - Fake the Source of Data	0.76	0.26	0.39
34 - HTTP Response Splitting	0.90	0.04	0.07
33 - HTTP Request Smuggling	0.82	0.07	0.13
weighted avg	0.90	0.71	0.77
Accuracy	0.70286653		

Судя по приведенным результатам классификации, метод Random Forest проявил себя несколько лучше, чем XGBoost. Однако не все классы были определены с одинаковой точностью. Например, для класса «153 - Input Data Manipulation» точность (Precision) составила 0.10. Однако итоговые показатели классификации достаточно хорошие. Это может быть связано с тем, что классы в наборе данных несбалансированные.

3.2.2. Подсчет букв, цифр и служебных символов

Данный подход основан на следующей идее: те поля, где присутствует большое количество аномальных символов, классификатор может распознать как атаку, так как их количество будет отлично от количества в полях нормального запроса.

Несмотря на то, что метод вычисления средней суммы ASCII символов показал довольно неплохие результаты, он малочувствителен к вариациям структуры веб-запроса: например, при удалении какого-либо символа средняя сумма ASCII может совсем незначительно изменится. Предложенный подход позволяет исключить этот недостаток путем контроля количества каждого символа.

Процедура предобработки заключается в следующем: подсчитывается количество букв, цифр и специальных символов для каждого поля веб-запроса. Затем в новую выборку для каждого поля исходного датафрейма записывается количество букв, цифр и специальных символов. Таким образом, будет сформирована новая выборка, число полей которой будет в 3 раза больше чем в исходной. На рисунке 3.3 представлен подробный алгоритм для предлагаемого метода кодирования полей веб-запроса.

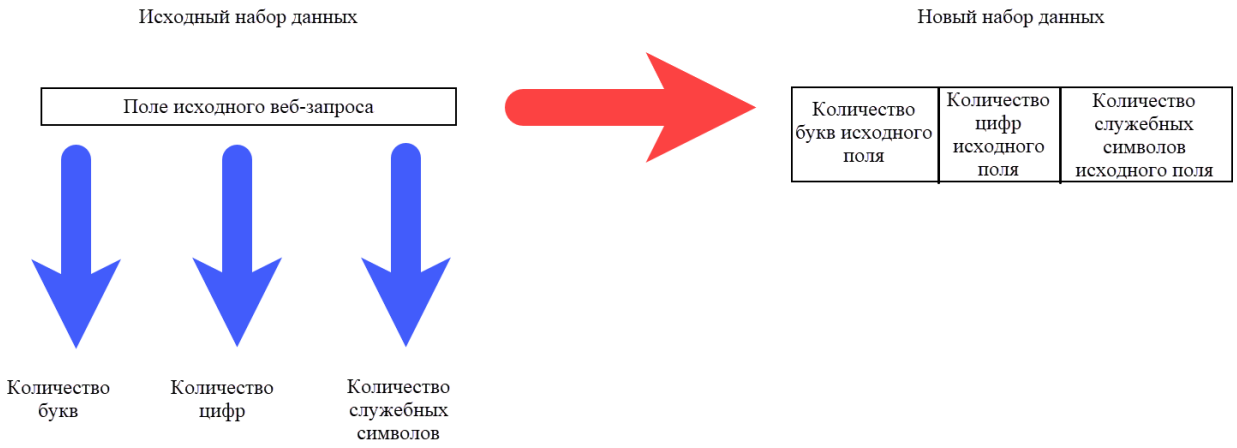

Рисунок 3.3 – Пример численного кодирования поля веб-запроса

После процедуры численного преобразования каждого поля веб-запроса необходимо очистить выборку от функций, которые не предоставляют полезной информации (неинформативные признаки). На рисунке 3.4 представлена информативность каждой функции поля веб-запроса.

Как можно увидеть, полезную информацию несет не только не только количество служебных символов. Количество букв и цифр некоторых полей веб-запроса также представляют большую важность для идентификации веб-атаки.

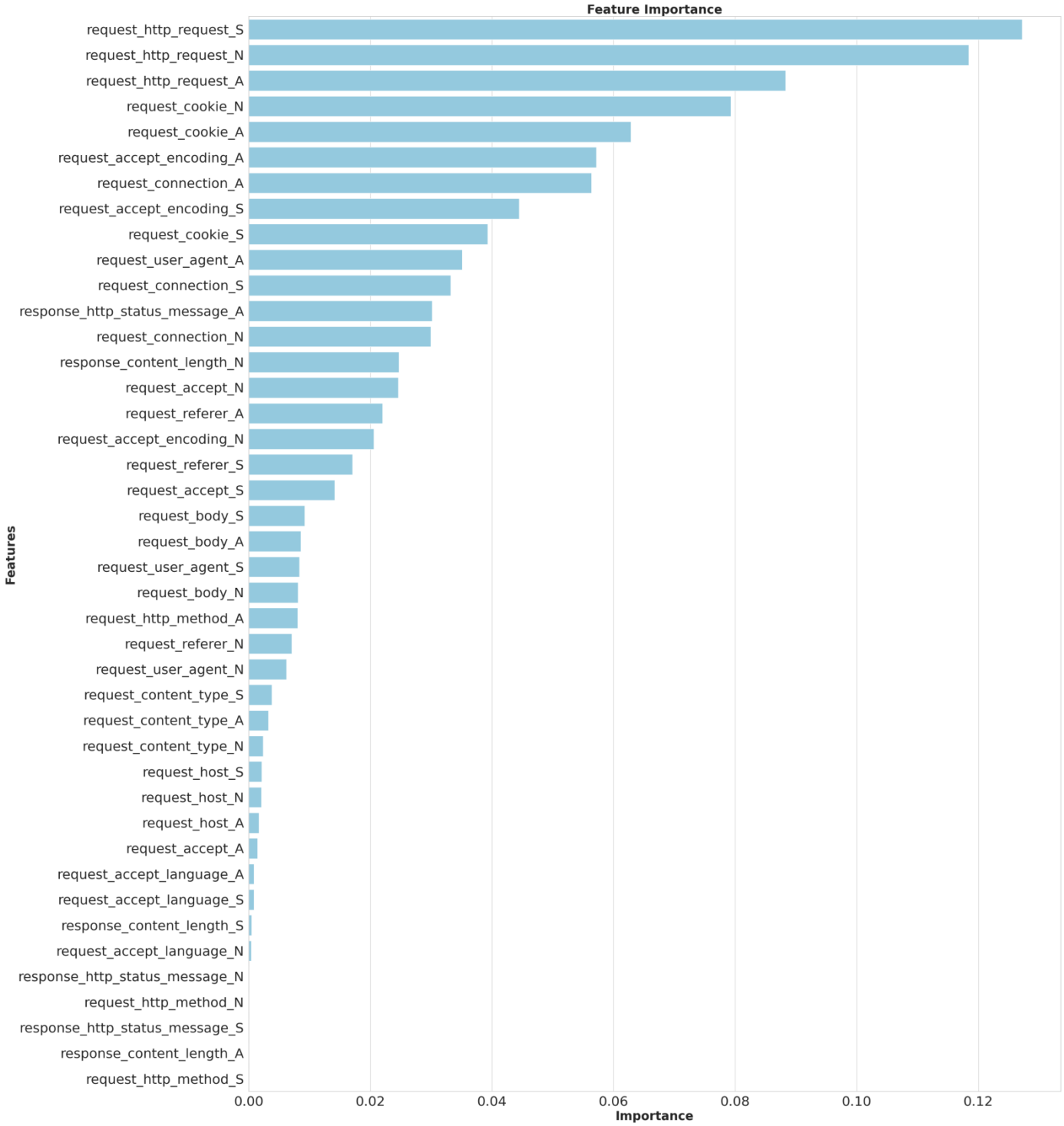

Рисунок 3.4 – Информативность признаков поля веб-запроса

В таблице 7 и таблице 8 приведены результаты классификации для выбранных показателей качества для 2 моделей: Random Forest и XGBoost соответственно.

Таблица 7 – Результаты классификации для вычисления количества букв, цифр и служебных символов для модели Random Forest

	Precision	**Recall**	**F1-score**
000 - Normal	0.97	0.97	0.97
272 - Protocol Manipulation	0.85	0.63	0.73
242 - Code Injection	1.00	0.96	0.98
88 - OS Command Injection	0.93	0.83	0.88
126 - Path Traversal	0.98	0.95	0.97
66 - SQL Injection	0.96	0.96	0.96
16 - Dictionary-based Password Attack	1.00	1.00	1.00
310 - Scanning for Vulnerable Software	1.00	1.00	1.00
153 - Input Data Manipulation	1.00	0.99	0.99
274 - HTTP Verb Tampering	0.97	0.98	0.97
194 - Fake the Source of Data	0.94	0.92	0.93
34 - HTTP Response Splitting	0.86	0.78	0.82
33 - HTTP Request Smuggling	0.96	0.95	0.96
weighted avg	0.96	0.95	0.96
Accuracy	0.954997		

Таблица 8 – Результаты классификации для вычисления суммы ASCII символов для модели Random Forest

	Precision	**Recall**	**F1-score**
000 - Normal	0.96	0.95	0.95
272 - Protocol Manipulation	0.94	0.55	0.70
242 - Code Injection	0.99	0.97	0.98
88 - OS Command Injection	0.94	0.83	0.88
126 - Path Traversal	0.98	0.92	0.95
66 - SQL Injection	0.94	0.92	0.93
16 - Dictionary-based Password Attack	1.00	0.99	1.00
310 - Scanning for Vulnerable Software	1.00	1.00	1.00

Продолжение таблицы 8

153 - Input Data Manipulation	0.99	0.98	0.99
274 - HTTP Verb Tampering	0.99	0.99	0.99
194 - Fake the Source of Data	0.90	0.85	0.88
34 - HTTP Response Splitting	0.93	0.75	0.83
33 - HTTP Request Smuggling	0.97	0.96	0.96
weighted avg	0.95	0.93	0.94
Accuracy	0.917771		

По таблице 7 и таблице 8 можно сказать, что данный метод предобработки данных полей веб-запроса проявил себя несколько лучше, чем метод вычисления средней суммы ASCII символов.

3.3. Проверка результатов на другом наборе данных

На предыдущих этапах исследования была разработана модель, которая позволяет получить хорошие результаты классификации на исходном наборе данных (SR-BH 2020). Но является ли она универсальной, то есть будет ли давать приемлемые результаты на другой выборке без дополнительного обучения на ней? Для ответа на этот вопрос использовался другой, независимый набор данных ECML/PKDD 2007 [12].

Набор данных ECML/PKDD 2007 (European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases) представляет собой один из популярных наборов данных, используемых в области машинного обучения и анализа данных. Этот набор данных был представлен на конференции ECML/PKDD в 2007 году. Набор данных ECML/PKDD 2007 содержит информацию о доменах веб-страниц и их признаках. Он предназначен для задачи классификации веб-запросов на два класса: нормальные и запросы с мошенническим содержимым. Набор данных ECML/PKDD 2007 был сгенерирован путем записи реального трафика, который затем был обработан для очистки информации. В этой выборке содержится информация о различных признаках веб-запросов, таких как URL, текстовое содержимое, ссылки, и другие характеристики. Каждая страница помечена меткой, указывающей на ее классификацию (нормальная или мошенническая).

Для начала была предпринята попытка обучить и протестировать обученный на данной выборке классификатор. В таблице 9 и таблице 10 приведены результаты классификации для методов вычисления средней суммы ASCII символов и подсчета количества букв, цифр и служебных символов соответственно. Модель классификации – Random Forest.

Таблица 9 – Результаты классификации для вычисления количества букв, цифр и служебных символов для модели Random Forest

	Precision	**Recall**	**F1-score**
Anomalous	0.81	0.87	0.84
Valid	0.89	0.83	0.86
weighted avg	0.85	0.85	0.85
Accuracy	0.851656		

Таблица 10 – Результаты классификации для вычисления суммы ASCII символов для модели Random Forest

	Precision	**Recall**	**F1-score**
Anomalous	0.59	0.84	0.69
Valid	0.79	0.51	0.62
weighted avg	0.70	0.66	0.65
Accuracy	0.659094		

Затем было принято решение проверить, насколько хорошие результаты можно получить с помощью модели, обученной на данных SR-BH 2020, при тестировании на наборе данных ECML/ PKDD 2007. Для этого предварительно необходимо сопоставить поля веб-запроса SR-BH 2020 с полями ECML/PKDD 2007, очистив неиспользуемые. Также тип классификации стал бинарным, так как в наборе данных ECML/PKDD 2007 присутствуют всего 2 метки класса («Valid» и «Anomalous»). Результаты классификации обученной модели представлены в таблице 11.

Таблица 11 – Результаты тестирования предварительно обученной модели для метода вычисления количества букв, цифр и служебных символов

	Precision	**Recall**	**F1-score**
Anomalous	0.44	0.76	0.55
Valid	0.47	0.18	0.26
weighted avg	0.46	0.44	0.40
Accuracy	0.444337		

Такие посредственные результаты могут быть обусловлены тем, что проверочная выборка может не в полном объеме отражать структуру и содержание веб-запросов. Поэтому было принято решение обучить и проверить модель Random Forest на объеденной выборке SR-BH 2020 и ECML/PKDD 2007. Ниже в таблице 12 приведены результаты классификации для объеденных выборок SR-BH 2020 и ECML / PKDD2007 в равном соотношении.

Таблица 12 – Результаты классификации на объединенной выборке для метода вычисления количества букв, цифр и служебных символов

	Precision	**Recall**	**F1-score**
Anomalous	0.87	0.87	0.87
Valid	0.90	0.90	0.90
weighted avg	0.89	0.89	0.89
Accuracy	0.888835		

На объединенной выборке результаты получились почти в 2 раза лучше, чем при тестировании на наборе данных ECML/ PKDD 2007. Это может говорить о том, что объединенная выборка в бóльшей степени отражает структуру веб-запросов. Однако, можно сделать вывод. что модель, обученная на наборе данных SR-BH 2020 не является универсальной.

ЗАКЛЮЧЕНИЕ

В данной работе были проведены исследования по выявлению веб-атак на основе HTTP-запросов. Была подробно изучена проблематика и актуальность данной задачи, разобраны особенности работы с веб-запросами. Также были предложены 2 способа численного кодирования полей веб-запросов для последующей multi-label классификации по шаблону CAPEC, и была проведена оценка качества нескольких моделей классификации.

Оба способа показали достаточно хорошие результаты, что может послужить в дальнейшем интеграцией модели в коммерческие средства защиты веб-приложений, которая может использоваться во всех типах сценариев с различными уровнями угрозы.

Однако, как показали проведенные исследования, модель классификатора, обученная на наборе данных SR-BH 2020, не является универсальной, то есть имеет ограничения в своей применимости. Причиной этому скорее всего является нерепрезентативность набора данных, на которых была обучена модель. Это означает, что данные недостаточно точно отражают широкий спектр вариаций, с которыми модель может столкнуться в реальных условиях.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Блог о защите от DDoS и кибербезопасности DDoS-Guard: [Электронный ресурс]. URL: https://ddos-guard.net/ru/blog?news=1&tag=website
2. The Hiscox Cyber Readiness Report: [Электронный ресурс]. URL: https://www.hiscox.co.uk/cyberreadiness
3. Спецификация протокола передачи данных HTTP: [Электронный ресурс]. URL: https://datatracker.ietf.org/doc/html/rfc2616
4. Web Application Firewall 3.0 Administration Guide: [Электронный ресурс]. URL: https://www.sonicwall.com/techdocs/pdf/waf-3-0-administration-guide.pdf
5. DATASHEET Runtime Application Self-Protection RASP: [Электронный ресурс]. URL: https://www.imperva.com/resources/datasheets/Runtime-Application-Self-Protection-RASP.pdf
6. I.V. Kotenko, I.B. Saenko, R.M. Yusupov. NEW GENERATION OF SECURITY INFORMATION AND EVENT MANAGEMENT SYSTEMS: [Электронный ресурс]. URL: https://infocom.spbstu.ru/userfiles/files/articles/2014/3/01.pdf
7. ENDPOINT DETECTON & RESPONSE: [Электронный ресурс]. URL: https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/docs/vmwcb-datasheet-edr.pdf
8. Practical Guide to Threat Hunting: [Электронный ресурс]. URL: https://www.threathunting.net/files/hunt-evil-practical-guide-threat-hunting.pdf
9. Классификация шаблонов атак CAPEC: [Электронный ресурс]. URL: https://capec.mitre.org/
10. N.A. Krivosheev, V.G. Spitsyn. Machine Learning Methods for Classification Textual Information: [Электронный ресурс]. URL: https://www.graphicon.ru/html/2019/papers/10/Volume1_paper_55.pdf
11. Tomás Sureda Riera, Juan-Ramón Bermejo Higuera. A new multi-label dataset for Web attacks CAPEC classification using machine learning techniques: [Электронный ресурс]. URL: https://www.sciencedirect.com/science/article/pii/S0167404822001833
12. Набор данных ECML/PKDD 2007; [Электронный ресурс]. URL: https://github.com/msudol/Web-Application-Attack-Datasets/tree/master/OriginalDataSets