****МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)»

**Институт № 4 «Радиоэлектроника, инфокоммуникации и информационная безопасность» Кафедра 402 Группа М4О-414Б-18 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Направление подготовки 10.03.02 \_\_\_\_\_\_\_ \_\_\_\_\_**

**Профиль \_\_\_\_\_**

**Квалификация бакалавр \_\_\_\_\_**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**БАКАЛАВРА**

На тему: Разработка средства защиты информации от несанкционированного доступа на основе модели поведения пользователя в веб-среде\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_

Автор ВКРБ \_\_\_\_\_\_\_\_\_Приданников Никита Денисович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество полностью) (подпись)

Руководитель \_\_\_\_\_\_\_\_Карпухин Евгений Олегович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество полностью) (подпись)

Консультант \_\_\_\_\_\_\_\_ Мешавкин Константин Викторович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество полностью) (подпись)

**К защите допустить**

Заведующий кафедрой \_\_402 Мазепа Роман Богданович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(№ каф.) (фамилия, имя, отчество полностью) (подпись)

\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

Москва 2022

**Оглавление**

[Введение 5](#_Toc103480798)

[1. Исследование модели поведения пользователя в веб-среде и поиск подходящих характеристик 10](#_Toc103480799)

[1.1 Использование поведенческой аутентификации пользователя как метод защиты информации 10](#_Toc103480800)

[1.2 Классификация биометрических характеристик пользователя 10](#_Toc103480801)

[1.2.1 Описание статических параметров личности 11](#_Toc103480802)

[1.2.2 Описание динамических параметров личности 13](#_Toc103480803)

[1.2.3 Сравнение биометрических характеристик 13](#_Toc103480804)

[1.3 Обзор методов идентификации пользователей веб-среды 15](#_Toc103480805)

[1.3.1 Описание идентификации личности с помощью анализа последовательности посещенных веб-сайтов 15](#_Toc103480806)

[1.3.2 Описание идентификации по действиям пользователя на сайте 16](#_Toc103480807)

[1.4 Вывод 17](#_Toc103480808)

[2. Обзор существующих средств защиты информации, исследующих шаблоны поведения человека в веб-среде 20](#_Toc103480809)

[2.1 Обзор ПО Plurilock DEFEND 20](#_Toc103480810)

[2.2 Обзор ПО BioCatch 20](#_Toc103480811)

[2.3 Обзор ПО KeyTrac 21](#_Toc103480812)

[2.4 Обзор ПО TypingDNA 23](#_Toc103480813)

[2.5 Вывод 24](#_Toc103480814)

[3. Формирование исходных данных на основе проведенного обзора для обучения алгоритмов искусственного интеллекта, их анализ и предобработка 26](#_Toc103480815)

[3.1 Сбор данных 26](#_Toc103480816)

[3.2 Преобразование собранных данных в набор характеристических признаков 27](#_Toc103480817)

[3.3 Шкалирование данных 29](#_Toc103480818)

[3.4 Вывод 32](#_Toc103480819)

[4. Обоснование и выбор алгоритмов для построения моделей машинного обучения и разработка системы 34](#_Toc103480820)

[4.1 Обзор применимых алгоритмов машинного обучения и выбор подходящего 34](#_Toc103480821)

[4.2 Подбор параметров модели машинного обучения на основе собранных данных 36](#_Toc103480822)

[4.3 Описание структуры разрабатываемого СЗИ 37](#_Toc103480823)

[4.3.1 Описание структуры и алгоритм работы сервера 38](#_Toc103480824)

[4.3.2 Описание структуры и алгоритм работы клиента 40](#_Toc103480825)

[4.4 Вывод 42](#_Toc103480826)

[5. Тестирование работы реализованной программы по выявлению НСД и оценивание полученных результатов 44](#_Toc103480827)

[5.1 Пользовательское тестирование программы 44](#_Toc103480828)

[5.2 Оценивание полученных результатов 46](#_Toc103480829)

[5.3 Вывод 47](#_Toc103480830)

[Заключение 49](#_Toc103480831)

[Список литературы 51](#_Toc103480832)

[Приложение 1 52](#_Toc103480833)

[Приложение 2 54](#_Toc103480834)

[Приложение 3 55](#_Toc103480835)

[Приложение 4 58](#_Toc103480836)

[Приложение 5 59](#_Toc103480837)

[Приложение 6 61](#_Toc103480838)

[Приложение 7 62](#_Toc103480839)

# Введение

С каждым годом информационные технологии развиваются все с большей скоростью. С распространением компьютеров и Интернета в повседневной жизни потребность в надежной информационной безопасности неуклонно растет, потому что вся важная информация хранится в компьютере пользователей и на серверах компаний. Информационная безопасность – это комплекс мер, направленный на обеспечение свойств информации: конфиденциальность, целостность и доступность. Из них наиболее важным является сохранение конфиденциальности, или предотвращение утечки информации. Киберпреступники с каждым днем разрабатывают новые методы для вторжений и кражи личных данных, поэтому непрерывная защита является ключевым фактором. Необходимо разрабатывать системы, позволяющие предотвращать попытки несанкционированного доступа к информации. Поэтому одной из основных задач системы информационной безопасности является идентификация и аутентификация любого субъекта доступа. Идентификация – это процесс распознавания субъекта по его идентификатору. Аутентификация – это подтверждение подлинности предъявленного оператором идентификатора.

Существующие системы защиты информации можно разделить на три категории по фактору аутентификация пользователя:

1. Аутентификация пользователя с использованием секретного информации (пароль, PIN, ключ). Преимуществом данных систем является простота использования. Однако они наименее надежны по сравнению с остальными, так как секретная информация может быть украдена или передана. Также часто пользователи используют одинаковые и ненадежные пароли, что делает их взлом еще проще.
2. Аутентификация пользователя с использованием физических объектов, принадлежащих пользователю (токены, магнитные карты). В данном методе злоумышленнику сложнее получить доступ к устройству аутентификации, однако и стоимость системы более высока.
3. Аутентификация пользователя по биометрике. Биометрика — это уникальные биологические или физиологические характеристики, присущие каждому человеку. Преимущество данных систем заключается в том, что данные признаки невозможно забыть или потерять. Однако такие системы должны обладать высокой чувствительностью.

Аутентификация может проводиться одноразово, например, при входе в систему. Такая методика называется статической аутентификацией. Однако у данного подхода существует значительный недостаток. После проверки идентификатора личность и деятельность пользователя никак не отслеживается. При динамической аутентификации пользователь проверяется на протяжении всей его работы в системе, что решает проблему первого метода. Даже при удачной первичной аутентификации с помощью токена и пароля злоумышленник не сможет завладеть компьютером. Данный метод позволяет предотвращать утечки информации с компьютеров сотрудников, которыми мог бы воспользоваться внутренний злоумышленник. Это позволяет повысить защищенность рабочих станций сотрудников, а также защищенность хранимой информации в организации в целом, т.к. с помощью рабочих станций сотрудников у злоумышленников появляется доступ к системе предприятия, а значит и доступ к коммерческим документам. Системы непрерывной аутентификации способны предотвратить подобного рода несанкционированный доступ в систему с помощью активной учетной записи сотрудников организации.

Работа большинства существующих систем основана на статической аутентификации, поэтому разработка системы, непрерывно проверяющей личность пользователя при работе в веб-среде, используя надежные виды идентификации, является актуальной проблемой на сегодняшний день.

**Объектом исследования** является цифровая активность пользователя в сети Интернет.

**Предметом исследования** являются виды биометрических и поведенческих данных, существующие средства зашиты информации, анализирующие эти данные, методы обработок данных и построения характеристических признаков для обучения моделей искусственного интеллекта, алгоритмы машинного обучения, а также способы построения клиент-серверного приложения.

Основной **целью** выпускной квалификационной работы является создание прототипа средства защиты информации от несанкционированного доступа с применением искусственного интеллекта, непрерывно аутентифицирующего пользователя по его поведению в веб-среде.

В соответствии с поставленной целью в работе решены следующие **задачи**:

1. Сформулированы объект исследования, предмет исследования, цель исследования и обоснована актуальность исследования.
2. Проведен обзорный анализ следов цифровой активности пользователя, оставляемых при работе в веб-среде.
3. Проведен обзорный анализ существующих разработок, решающих задачи поведенческой аутентификации пользователя.
4. Реализовано приложение, позволяющее собирать данные пользователя о его работе с клавиатурой.
5. Рассмотрены характеристические признаки, которые можно получить из собранных данных для анализа поведения пользователя.
6. Реализован модуль для преобразования событий в характеристические признаки.
7. Рассмотрены применимые алгоритмы машинного обучения и выбран наиболее подходящий из них, подобраны параметры для модели.
8. Разработан прототип средство защиты информации на основе клиент-серверного взаимодействия, включающее в себя на стороне сервера выбранную модель для анализа данных, передаваемых от клиента, и непрерывно анализирующее поведение пользователя.
9. Протестирована работа созданного СЗИ и оценены результаты.

Список основных результатов, выносимых на защиту:

1. Формулировки объекта исследования, предмета исследования, цели исследования и обоснования актуальности выпускной квалификационной работы.
2. Результат обзора поведенческих характеристик человека, позволяющих идентифицировать личность в сети Интернет.
3. Результат обзора существующих средств защиты информации, анализирующих поведение пользователя и проводящих непрерывную аутентификацию.
4. Результат сбора необходимых для обучения данных и их преобразование в характеристические признаки.
5. Результат создание средства защиты информации от несанкционированного доступа на основе непрерывной аутентификации пользователя по его поведению в веб-среде.
6. Результат оценивания работы созданного программного обеспечения на основе проведенных тестов.

**Научная новизна работы.** Научная новизна полученных результатов заключается в разработке веб-приложения, позволяющего собирать и анализировать данные, а также проводить непрерывную аутентификацию личности с применением технологий машинного обучения.

**Достоверность** полученных результатов обеспечивается научной обоснованностью приводимых исследований.

**Результаты внедрения.** Результаты выпускной квалификационной работы успешно опробованы и применены на практике.

**Структура выпускной квалификационной работы.** Выпускная квалификационная работа состоит из введения, пяти глав, заключения, списка использованных источников и 7 приложения. Общий объем выпускной квалификационной работы составляет 63 страниц машинописного текста. Библиографический список содержит 6 наименований.

**Краткое содержание выпускной квалификационной работы.**

В первой главе рассмотрены наиболее популярные методы идентификации личности в сети Интернет, их преимущества и недостатки, а также выбран признак, на основе которого разработанная система будет производить аутентификацию пользователя.

Во второй главе рассмотрены существующие средства защиты информации, принципы их работы, сферы применения и возможности, протестирован алгоритм работы одного из них.

В третьей главе реализован сбор поведенческих данных пользователя, рассмотрены характеристики, которые могут быть получены из этих данных, а также произведена подготовка данных в качестве обучающей выборки.

В четвертой главе рассмотрены возможные алгоритмы машинного обучения для реализации задачи, выбран наиболее подходящий из них, проведен эксперимент для подбора параметров модели на основе данных, собранных в главе 3, а также реализовано программное обеспечение, состоящее из сервера и клиента и проводящее непрерывную аутентификацию пользователя.

В пятой главе проведено тестирование созданного средства защиты информации и оценены результаты его работы, а также проанализированы преимущества и недостатки выбранного метода аутентификации.

# 1. Исследование модели поведения пользователя в веб-среде и поиск подходящих характеристик

## 1.1 Использование поведенческой аутентификации пользователя как метод защиты информации

Проблема идентификации пользователей в сети Интернет является одной из фундаментальных проблем киберпространства. Системы парольной аутентификации, которые наиболее распространены в повседневной жизни, обладают существенными недостатками: проводят аутентификацию единожды при входе в систему, при разглашении пароля любым способом система защиты становится абсолютно уязвима, а многие сайты хранят пароли в своих базах данных в открытом виде, что делает утечки личных данных еще опаснее. Методы поведенческой аутентификации позволяют решить данные проблемы с помощью технологий аутентификации пользователей с применением уникальных и естественных поведенческих характеристик, постоянно оценивая взаимодействие пользователя со своим компьютером в режиме реального времени.

Также аутентификация пользователей по поведению способна решать новые задачи, недоступные для классических систем аутентификации:

1. Программные обеспечения способны выявлять поведенческие аномалии, указывающие на нечеловеческие киберугрозы, такие как атаки с помощью ботов и вредоносного ПО.
2. Обнаруживать мошенничества с помощью социальной инженерии. Искусственный интеллект выявляет тонкие поведенческие индикаторы, которые позволяют предположить, что жертва невольно совершает действия.

## 1.2 Классификация биометрических характеристик пользователя

Набор биометрических признаков каждого человека уникален. В отличие от паролей и токенов, биометрические данные не могут быть потеряны, украдены или подслушаны, поэтому они представляют надежный способ определения идентичности человека. При использовании в сочетании с методами, построенными на том, что человек знает (например, пароль) или что он имеет (например, удостоверение личности), они обеспечивают дополнительный уровень безопасности. Биометрические характеристики делятся на две группы: статические и динамические или поведенческие признаки.

К первой группе относятся такие характеристики, которые заложены в каждом человеке от рождения и не меняются с течением времени или меняются незначительно. Однако из-за неизменности характеристик у них есть существенный недостаток – их проще подделать, чем динамические признаки.

Ко второй группе относятся признаки, основанные на поведенческих особенностях человека, таких как голос или ритм печати на клавиатуре.

Поведенческая биометрия дает ряд преимуществ по сравнению со статическими биометрическими технологиями. Их можно собирать ненавязчиво или даже без ведома пользователя. Сбор поведенческих данных часто не требует какого-либо специального оборудования и поэтому является очень экономически эффективным. Хотя большинство поведенческих биометрических данных не являются достаточно уникальными, чтобы обеспечить надежную аутентификацию человека, исследования показывают, что они обеспечивают достаточно высокую точность проверки личности. Недостатком может являться изменчивость признаков, что делает необходимым постоянное обновление эталонной модели.

### 1.2.1 Описание статических параметров личности

Наиболее популярными методами идентификации, основанными на статических параметрах, являются:

1. Идентификация по отпечатку пальца. Является одной из самых популярных технологий идентификации и аутентификации пользователя. Метод основан на анализе рисунка папиллярных узоров на пальцах людей, который является уникальным для каждого человека. Сначала с помощью специального сканера снимается отпечаток, затем он преобразовывается в цифровой код, который является эталоном для последующих процессов аутентификации. Преимуществом этого метода можно назвать удобство, надежность и достаточно простая интеграция в существующую систему безопасности.
2. Идентификация по сетчатке глаза. В данном случае анализируется рисунок кровеносных сосудов глазного дна. При просмотре на специальную удаленную световую точку глазное дно подсвечивается, что позволяет отсканировать этот рисунок с помощью специальной камеры.
3. Идентификация по радужной оболочке глаза. Как и в предыдущем методе, для сканирования радужной оболочки глаза, которая также является уникальной характеристикой каждого человека, необходима камера со специальным программным обеспечением. Цифровой код для идентификации получается следующим образом: сначала захватывается лицо пользователя, затем из изображения глаза выделяется рисунок радужной оболочки.
4. Идентификация по геометрии руки. С помощью камер и специальных подсвечивающих диодов считываются разные проекции ладони, по которым строится трехмерная модель кисти руки. Далее формируется свертки и происходит идентификация.
5. Идентификация по форме лица. Данный метод основан на построении трехмерного образа лица человека. Анализируются части лица, такие как брови, глаза, нос и губы, рассчитывается расстояние между ними, а затем строится уникальный шаблон, соответствующий конкретному человеку, на основе 12-40 характерных участков. Технологию возможно использовать с обычной видеокамерой, встроенной в персональный компьютер, однако для лучших результатов необходимо использовать специальные 3D-видеокамеры, позволяющие получить более точные данные.

Менее популярным, но заслуживающим отдельное внимание является метод идентификации по ДНК. Среди всех методов данный способ является самым точным для аутентификации и наиболее устойчив к подделке, однако существующие методы обработки ДНК занимают слишком много времени, а для анализа нужно специальное дорогостоящее оборудование, что делает данные метод применимым только для специальных экспертиз.

### 1.2.2 Описание динамических параметров личности

К поведенческим параметрам личности относят такие характеристики как:

1. Идентификация по рукописному почерку. Каждый человек обладает уникальным движением руки во время письма. Для измерения характеристик используют цифровую ручку или стилус, а также восприимчивый к давлению планшет. Сравнение происходит при написании одинаковой контрольной фразы или подписи человека. Затем происходит анализ либо самого написанного фрагмента, либо временных и статических характеристик написания.
2. Идентификация по динамике нажатий клавиш. В процессе печати на клавиатуре человек имеет уникальные динамические характеристики, описывающие его взаимодействие и динамику набора текста. Из преимуществ данного метода можно выделить отсутствие специального оборудования для записи информации.
3. Идентификация по голосу. Несомненным преимуществом метода идентификации, основанного на анализе голосовых параметров пользователя, является простота реализации, так как необходим лишь микрофон. Поэтому он чаще других динамических параметров встречается в повседневной жизни, например, для клиентов банковских учреждений. В качестве признаков из полученных голосовых данных извлекаются следующие характеристики: частотные и статистические параметры, интонация, высота тона и другие. Серьезным недостатком этого метода является низкая точность идентификации из-за существенных отличий голоса при различном физическом и эмоциональном состоянии человека.

### 1.2.3 Сравнение биометрических характеристик

Рассмотренные биометрические признаки имеют свои особенности, преимущества и недостатки. Для более удобного сравнения, а также выбора наиболее подходящего варианта составим таблицу 1. Все относительные характеристики получены на основе имеющихся исследований [[1, 2](#_Список_литературы)].

Таблица . Сравнение биометрических характеристик

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Признак | Устойчивость к внешним факторам | Стоимость оборудования и сложность измерения | Устойчивость к подделке | Доп. оборудование |
| Отпечаток пальца | Высокая | Средняя | Средняя | Да |
| Сетчатка глаза | Высокая | Средняя | Средняя | Да |
| Радужная оболочка глаза | Высокая | Средняя | Средняя | Да |
| Геометрия руки | Высокая | Средняя | Средняя | Да |
| Форма лица | Средняя | Средняя | Средняя | Да |
| ДНК | Высокая | Высокая | Высокая | Да |
| Рукописный почерку | Средняя | Средняя | Средняя | Да |
| Динамика нажатий клавиш | Средняя | Низкая | Средняя | Нет |
| Голос | Низкая | Низкая | Низкая | Нет |

Исходя из таблицы среди биометрических признаков наилучшими показателями обладает метод идентификации на основе ДНК, однако при текущем уровне развития технологий для его использования необходимы большие финансовые и временные затраты.

Сравнивая признаки по столбцу «Доп. оборудование» видно, что методов, которые можно использоваться массово для аутентификации пользователя в сети Интернет два – голос и динамика нажатий клавиш. Также преимуществами двух этих признаков является возможность непрерывного и незаметного сбора данных.

## 1.3 Обзор методов идентификации пользователей веб-среды

Существует такая область исследований как Web Mining – использование методов интеллектуального анализа данных и машинного обучения для выявлений закономерностей в сети Интернет. Одной из ее частью является анализ использования веб-ресурсов, заключающийся в автоматическом обнаружении шаблонов в маршруте передвижения пользователя или его взаимодействии с веб-ресурсами. Получение уникального поведенческого следа, основанного на анализе шаблона использования веб-приложения для конкретных пользователей или группы пользователей, позволяет создать для каждого человека его модель поведения и выявлять аномальные действия для защиты от взломов.

### 1.3.1 Описание идентификации личности с помощью анализа последовательности посещенных веб-сайтов

Поскольку все больше людей используют Интернет для общения, работы и развлечений, существуют подходы, позволяющие аутентифицировать пользователя, основываясь на его истории просмотра веб-страниц. По мере того, как взаимодействие с Интернетом становится более естественным, анализ поведения при просмотре веб-страниц может выявить шаблоны, подходящие для идентификации пользователя. Собирая лишь метку времени, а также URL-адрес, можно получить большое количество характеристических признаков, по которым искусственный интеллект может различать пользователей между собой. Информация о посещениях веб-ресурсов группируется на сеансы или сессии. Сеанс считается завершенным, если пользователь не посещал како-либо ресурс в течение 30 минут. Анализировать можно как и отдельно сессии между собой, так и переходы между сайтами внутри одной сессии. Исходя из этого, можно получить различные параметры для сравнения пользователей. При исследовании сессий рассматриваются распределения дней недели и времен суток. Так как в настоящее время большое количество сотрудников работают по гибкому графику или удаленно, эти распределения будут отличаться у разных людей. У одного пользователя может отсутствовать активность по выходным дням. Другой пользователь может быть активен в основном утром, а другой наоборот – ближе к вечеру. Также можно высчитывать такие характеристики как общее число просмотренных страниц, среднюю продолжительность просмотра одной страницы и количество уникальных страниц. При анализе одного сеанса фиксируются характеристики просмотра страниц. К ним относятся:

* Паузы – это время, проведенное пользователем на веб-странице. Вычисляется как разность временных меток двух последовательно просмотренных страниц. Исследования показывают, что данная характеристика следует степенному закону распределения. Следовательно, можно использовать экспоненциальную функцию в качестве сигнатуры для сравнения промежуточных распределений.
* Время между повторными посещениями одной страницы. Скорость повторного посещения также может служить показателем идентичности пользователя.
* Категории посещенных сайтов. Кодирую сайты по семантическому и стилистическому содержанию. Примерами таких категорий являются блог, социальная сеть, статьи. По данному признаку также можно классифицировать пользователей и выявлять аномалии в посещенных сайтах.

Недостатком этого метода является длительный сбор данных для обучения – проведенные исследования [[3](#_Список_литературы)] основываются на месяце работы в браузере. Также авторы исследований утверждают, что особенности поведения при просмотре веб-страниц не являются достаточно индивидуальными для полноценной аутентификации, однако они могут служить хорошим дополнением к существующей системе.

### 1.3.2 Описание идентификации по действиям пользователя на сайте

Сайты и веб-приложения предоставляют посетителям и пользователем различные способы взаимодействия с ресурсом, поэтому, анализируя данные о пользовательских сценариях взаимодействия с контентом и их поведения, можно получить дополнительные признаки для идентификации. Недостаток данного метода в том, что он применим лишь в качестве одного компонента существующего веб-сайта и не актуален для защиты корпоративной локальной сети от несанкционированного доступа, в отличие от предыдущего метода. Так, например, исследования компании «Яндекс» показали [[4](#_Список_литературы)], что возможно отличить взломщика от легитимного владельца аккаунта почты. Каждый человек пользуется почтой по-разному: некоторые люди работают по ночам и отправляют множество электронных писем на разные адреса; другие проверяют почту только утром и переписываются только с одним или двумя людьми. Все эти особенности могут быть использованы для создания поведенческого профиля, который может служить поведенческой биометрией для человека. Длина электронных писем, время дня отправки почты, частота очистки почтового ящика и адреса получателей могут быть объединены для создания базового вектора, описывающего поведение человека в электронной почте. Строя профили поведения пользователей, возможно определить дату и время скачка показателей, что может упростить поиск места взлома.

## 1.4 Вывод

Проанализировав существующие биометрические и поведенческие характеристики, можно заключить, что существует большое множество параметров, по которым можно определить личность пользователя. Используя биометрические признаки, пользователь может быть определен по его статических или динамическим признакам. Статические признаки – те признаки человека, которые даются ему с рождения и в течение жизни не изменяются или меняются незначительно. Такие признаки являются уникальными для человека и могут быть использованы в качестве методов аутентификации личности. Но этот подход подразумевает под собой использование биометрических данных в качестве начальной точки входа в систему и в процессе работы в системе подтверждать эти данные через определенный промежуток времени будет крайне неудобно.

Исследование показало, что для постоянной и непрерывной аутентификации подходит использование динамический параметров человека. Однако, динамические параметры отличаются от статической своей неустойчивости к различным изменениям характеристик сотрудника. Поэтому эталонная модель динамических биометрических данных должна будет обновляться в системе организации с целью получения актуальных динамических характеристик биометрических данных человека. Исходя из данных таблицы 1, можно заметить еще один плюс динамических характеристик: для некоторых из них не нужно использовать дополнительное оборудование для снятия динамических параметров человека. Это преимущество позволяет снизить затраты на покупку дорогостоящего оборудования, что позволит сэкономить на затратах. Выбирая параметр аутентификации личности выбор стоит между аутентификации с помощью голоса или клавиатурного почерка. Исследование показывает, что параметр Голос является неустойчивой характеристикой человека, так как может корректироваться в течение жизни, а также изменяться из-за болезней или разных психологических состояний. Клавиатурный почерк, в свою же очередь, обладает более высокой устойчивостью к внешним факторам.

В дополнение был предложен метод непрерывной аутентификации с помощью поведения человека в веб-среде. Были разработаны параметры, по которым можно аутентифицировать человека. Анализ показал, что данный подход является не устойчивым к внешним изменениям и не может применяться в качестве метода аутентификации, т.к. поведение человека в веб-среде зависит от многих факторов: от его целей поиска в веб-среде, его нестабильности в режиме работы, активности на различных интернет-ресурсах в зависимости от важности поставленных задач и т.д. Поэтому можно сделать вывод, что метод анализа поведения человека в веб-среде нельзя использовать в качестве параметра непрерывной аутентификации сотрудника на рабочей станции.

Рассмотренные поведенческие признаки, оставляемые пользователем во время работы в сети Интернет, не являются достаточно надежными для использования в качестве основных характеристических черт поведения по сравнению с биометрическими, однако могут использоваться в качестве дополнительных факторов идентификации. Поэтому в качестве основного метода аутентификации в разрабатываемом СЗИ выбрана динамика нажатия клавиш клавиатуры.

# 2. Обзор существующих средств защиты информации, исследующих шаблоны поведения человека в веб-среде

## 2.1 Обзор ПО Plurilock DEFEND

Plurilock предоставляет решения для невидимой многофакторной аутентификации и непрерывной аутентификации с использованием поведенческих биометрических технологий и технологий машинного обучения. Для того чтобы с высокой точностью аутентифицировать пользователя, приложению необходимо обучиться на порядка 30 минутах его непрерывной работы с клавиатурой персонального компьютера. Помимо этого, в ходе взаимодействия пользователя с машиной для улучшения качества модели анализируется его работа с мышью. Когда движения мыши и частота нажатия клавиш пользователя соответствуют их обычному поведению, журналы отражают незначительный риск или не отражают его вообще. Когда движения клавиатуры и указателя начинают казаться системе необычными, в журнал заносятся события средней степени риска. При превышении настраиваемого порога риска DEFEND уведомляет сотрудников службы безопасности и регистрирует незаконные действия. Также платформа периодически дообучает модель, чтобы учитывать изменение динамики его работы с устройствами в течение длительного промежутка времени. Подтверждение личности пользователя происходит каждые 3–5 секунд в течение всего времени работы. Данное решение может работать как и отдельный компонент, так и встраиваться в готовое приложение. Продукт поддерживается всеми популярными сегодня ОС.

## 2.2 Обзор ПО BioCatch

Еще одним решением, осуществляющим динамическую многофакторную аутентификацию пользователей, является Biocatch. В отличие от традиционных средств аутентификации и проверки личности, платформа пассивно определяет, является ли пользователь подлинным в течение всего сеанса, а не только в одной точке цифрового потока, такой как вход в систему. BioCatch непрерывно отслеживает физическое и когнитивное поведение пользователя от входа в систему до выхода из нее. Программное обеспечение анализирует различные виды цифровой активности, включая движение мыши (скорость, модели движения и прокрутки колеса), динамику нажатия клавиш, работу с сенсорным экраном (площадь зоны взаимодействия, а также сила нажатия), а также ориентации устройства в пространстве. Данный продукт предлагается использовать для защиты web-приложений от несанкционированного доступа и для защиты пользователей онлайн банковских систем от доступа злоумышленников к их счетам. Также есть возможность интеграции в мобильное приложение для IOS и Android. Аналитикам предоставляется широкий инструмент мониторинга, отображающий на различных графиках возможные риски и аномалии, а также формирующий информативные отчеты о работе системы.

## 2.3 Обзор ПО KeyTrac

Продукт KeyTrac является решением, позволяющим осуществлять как идентификацию, так и аутентификацию пользователей в фоновом режиме именно на основе динамики нажатия клавиш. KeyTrac предназначен для встраивания в собственное приложение и анализирует только поля, к которым у него есть доступ, и не может записывать данные, введенные вне этих настроенных текстовых полей. Сначала происходит обучение модели с помощью специального компонента KeyTrac Recorder, который в последствие и отвечает за захват данных о нажатиях клавиш. Затем с помощью специально разработанного KeyTrac API данные передаются на сервер, который вычисляет оценку совпадения с данными в профиле пользователя, а после вычисления модель рекомендует доверять человеку или нет. При этом данные, хранящиеся на сервере, полностью анонимизируются. В добавок, по соображениям безопасности, API KeyTrac доступен только через защищенное SSL HTTPS-соединение. Незашифрованные HTTP-соединения не поддерживаются. Решение отличается простой интеграцией в приложение, так как использует стандарт RESTful API для встраивания в собственное программное обеспечение. KeyTrac постоянно учится приспосабливаться к незначительным изменениям в стиле набора текста пользователем. Разработчики системы утверждают, что их решение нечувствительно к смене используемого языка ввода, а также к смене используемого оборудования. Одной из интересных функций именно этой платформы является обратный поиск, который помогает идентифицировать личность, основываясь только на ее динамике нажатия клавиш.

На сайте программы присутствует тестовая веб-версия сервиса, демонстрирующая работу алгоритма. Сначала предлагается ввести два небольших текста длинной 200-250 символов. Пример процесса обучения представлен на рисунке 1:

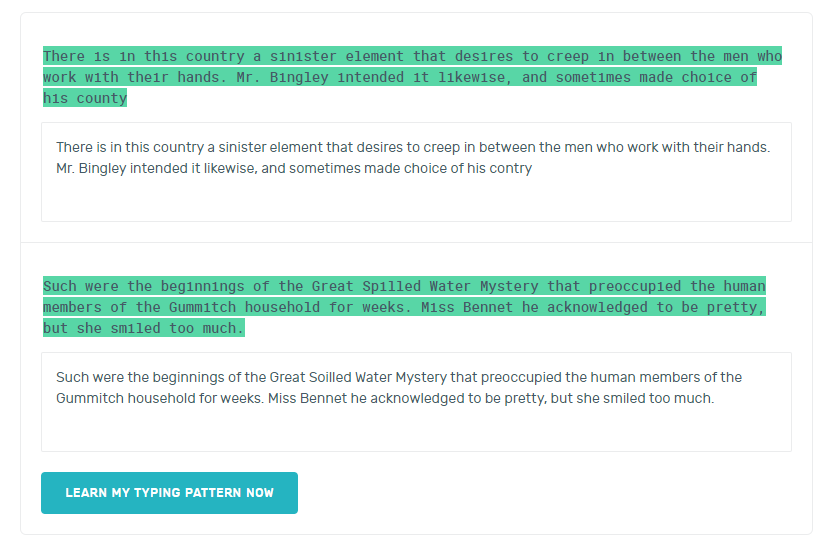


Рисунок 1. Пример обучения решения KeyTrac

После прохождения процесса обучения предлагается еще раз ввести один текст для проведения аутентификации. Процесс аутентификации представлен на рисунке 2:

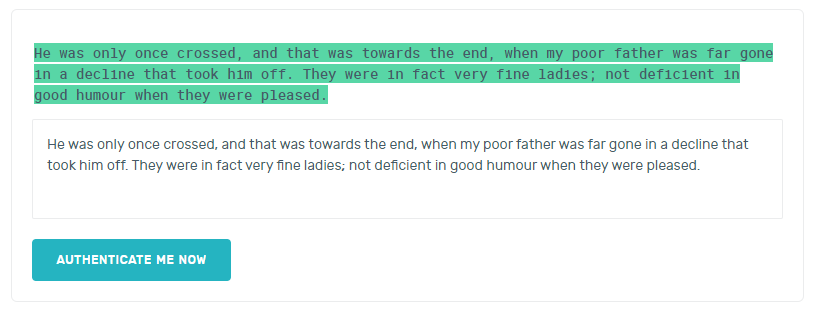


Рисунок 2. Пример аутентификации решения KeyTrac

После прохождения аутентификации алгоритм рассчитает вероятность принадлежности запроса к легитимному пользователю. Результат проведенного эксперимента показан на рисунке 3:

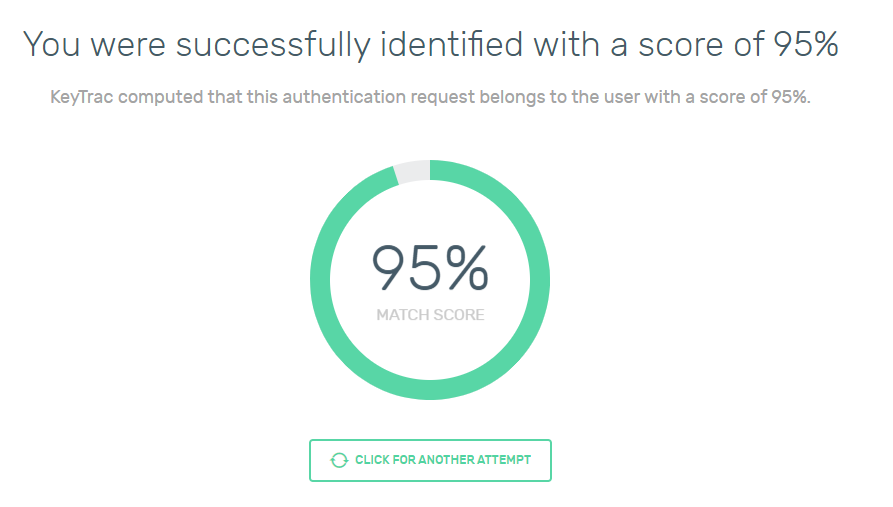


Рисунок . Вывод результата аутентификации

## 2.4 Обзор ПО TypingDNA

Одним из новых программных обеспечений, осуществляющих аутентификацию с помощью анализа клавиатурного почерка, является TypingDNA. Компания предоставляет пакет продуктов как для непрерывной (ActiveLock), так и для двухфакторной аутентификации, а также разработанное API для встраивания продукта в свое приложение. Решение соответствует зарубежным стандартам безопасности и применяет в государственных учреждениях и банковской сфере. Технология на основе ИИ анализирует шаблоны набора текста прямо на устройстве, поэтому данные о наборе текста сотрудника остаются конфиденциальными и никогда не покидают компьютера и не хранятся в облаке. При обнаружении злоумышленника программа незамедлительно автоматически блокирует персональный компьютер пользователя. Программа использует модель безопасности с нулевым доверием и предотвращает даже совместное использование устройства доверенным лицом, гарантируя, что каждый человек перед компьютером всегда является истинным авторизованным пользователем. Это важно при удаленной работе из дома, когда профессиональные задачи выполняются на личном компьютере, к которому имеют доступ все члены семьи, что может нанести вред компании. Как и предыдущие решения, программа работает в фоновом режиме, не влияя на работу сотрудников.

## 2.5 Вывод

В данной главе были рассмотрены существующие на сегодняшний день индустриальные решения, проводящие непрерывную аутентификацию пользователя. Динамическая аутентификация осуществляется при помощи использования записи биометрических данных, в том числе клавиатурного почерка. Программы осуществляют первичный сбор информации. Эта информация становится эталоном сравнения. После чего все новые приходящие данные сравниваются с эталоном и на основании внутренних алгоритмов вычисляют принадлежность биометрии клавиатурного почерка или других признаков к определенному лицу. В добавок, одно из данных приложений способно осуществлять обратный поиск, т.е., имея в доступности исключительно клавиатурный почерк сотрудника, программы смогут вычислить кем является данный сотрудник. Все программы отличаются простотой внедрения в системы различных организаций, а также отдельно производители этих программ выделяют модульность данных приложений, т.е. запись и оценку результатов осуществляют отдельные блоки программ. Также, для защиты биометрических данных пользователей некоторых из них требуется защищенное соединение между клиентом и сервером.

Суммируя все вышесказанное, можно подвести итог о том, что уже сегодня созданы ряд системы непрерывной аутентификации пользователя, которые позволяют идентифицировать пользователя по его поведению за компьютером, что включает в себя клавиатурный почерк и движение мыши. Непрерывная аутентификация позволяет защитить пользователя от использования его компьютера посторонним лицом в корыстных целях и выдать действия злоумышленника за действия сотрудника на данном компьютере. Однако все рассмотренные решения являются разработками зарубежных компаний, поэтому их применение в качестве средства защиты информации в отечественной коммерческой деятельность, а также в критической информационной инфраструктуре невозможно.

# 3. Формирование исходных данных на основе проведенного обзора для обучения алгоритмов искусственного интеллекта, их анализ и предобработка

## 3.1 Сбор данных

Первым этапом создания любой модели искусственного интеллекта является сбор данных, на основе которых будет произведено обучение для дальнейшего предсказания. Для анализа поведения пользователя при работе с клавиатурой и проведения практического исследования собираются следующие данные:

* Код клавиши
* Событие (нажатие/отжатие)
* Время произошедшего события в секундах

Собираться будут все клавиши, кроме специальных, таких как: Shift, Ctrl, f1-f12 и так далее. Для логирования был написан скрипт на языке Python. Для записи событий нажатия и отжатия клавиш использовалась библиотека pynput. Она специально предназначена для контролирования устройств ввода, а именно клавиатуры и мыши. Полный код скрипта представлен в Приложении 1. Ниже представлена таблица 2 соответствия клавиши клавиатуры записываемому коду клавиши.

Таблица 2. Соответствие клавиши и ее кода

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код клавиши | 81 | 87 | 69 | 82 | 84 | 89 | 85 | 73 | 79 | 80 |
| Клавиша | q – й | w – ц | e – у | r – к | t – е | y – н | u – г | i – ш | o – щ | p – з |
| Код клавиши | 219 | 221 | 65 | 83 | 68 | 70 | 71 | 72 | 74 | 75 |
| Клавиша | [ – х | ] – ъ | a – ф | s – ы | d – в | f – а | g – п | h – р | j – о | k – л |
| Код клавиши | 76 | 186 | 192 | 90 | 88 | 67 | 86 | 66 | 78 | 77 |
| Клавиша | l – д | ; – ж | ‘ – э | z – я | x – ч | c – с | v – м | b – и | n – т | m – ь |
| Код клавиши | 188 | 190 | 191 | 49 | 50 | 51 | 52 | 53 | 54 | 55 |
| Клавиша | , – б | . – ю | / – . | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Код клавиши | 56 | 57 | 48 | 189 | 187 | 220 | 192 |
| Клавиша | 8 | 9 | 0 | - | = | \ | ` – ё |

Группе учащихся было предложено ввести текст на английском языке, примеры текста взяты с сайта компании KeyTrac. Каждый участник исследования использовал свой персональный компьютер и клавиатуру. Всего в исследовании приняло участие 10 человек. Результат работы записывался в файл csv-формата и имел вид, представленный в таблице 3:

Таблица 3. Пример таблицы с собранными данными

|  |  |  |
| --- | --- | --- |
| keycode | event | time |
| 66 | Down | 1649610927.5280511 |
| 66 | Up | 1649610927.5757809 |
| 76 | Down | 1649610928.0806885 |
| 76 | Up | 1649610928.1571903 |
| … | … | … |

В последствие эти данные используются для тестирования и подбора нужных параметров модели.

## 3.2 Преобразование собранных данных в набор характеристических признаков

Собранные данные сами по себе не могут быть использованы напрямую для обучения модели. Их необходимо преобразовать в набор характеристических признаков, на основе которых и будет производиться аутентификация пользователя. Анализ динамики работы на клавиатуре основан на вычислении времени удержания клавиши (период, в течение которого клавиша находится в нажатом положении), а также продолжительностей перескоков между клавишами. На рисунке 4 представлены вычисляемые характеристики:

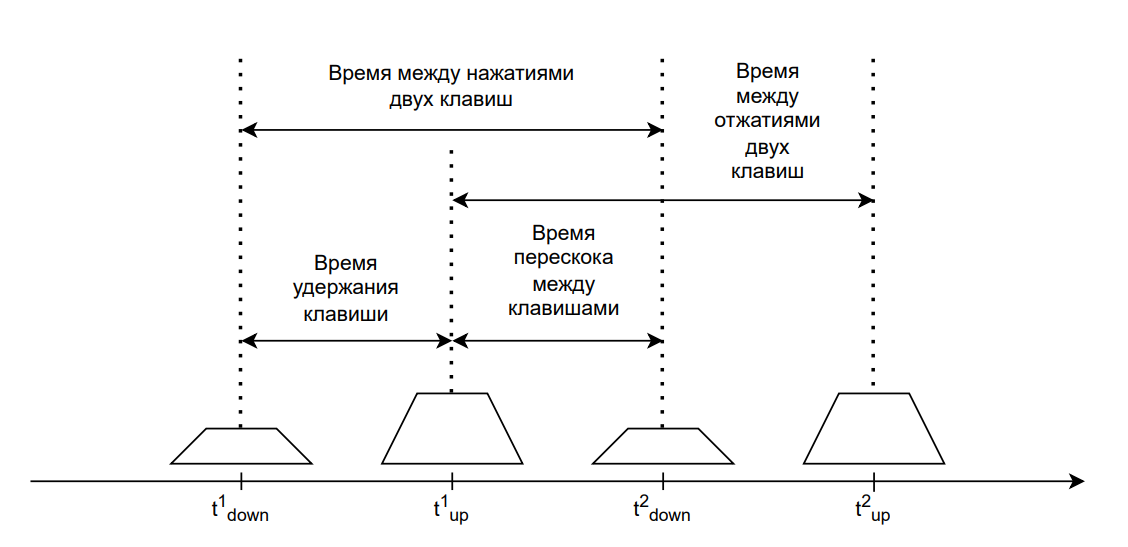


Рисунок . Вычисляемые характеристики

В данном случае вычисляются следующий параметры:

1. – промежутки времени удержания клавиши
2. – промежутки времени перескока между клавишами
3. – промежутки времени между отжатиями двух клавиш
4. – промежутки времени между нажатиями двух клавиш

Для преобразования данных реализован специальный модуль на языке Python, включающий в себя набор функций для обработки входных данных и сохранения этих данных в файл формата csv, а также чтения из файла. Преобразование данных происходит с помощью библиотеки Pandas, предназначенной для анализа структурных данных. Полный код модуля представлен в Приложении 2. Описание функций модуля представлены в таблице 4:

Таблица 4. Список функций модуля преобразования и их описание

|  |  |
| --- | --- |
| Название функции | Описание функции |
| convert\_json(data) | Преобразовывает входные данные в формате JSON в объект DataFrame |
| save\_capture\_data\_to\_file(user, data) | Сохраняет собранные с клавиатуры данные в файл с названием в формате «<имя\_пользователя>\_logs.csv» |
| read\_capture\_data\_from\_file(user) | Считывает сохраненные в файл события работы с клавиатурой |
| save\_processed\_data\_to\_file(user, data) | Сохраняет данные после преобразования в характеристические признаки в файл с названием в формате «<имя\_пользователя>\_processed\_data.csv» |
| processing\_keys(data) | Преобразовывает входной набор событий клавиатуры в формате DataFrame в набор характеристических признаков. Возвращает данные также в формате DataFrame |

Фрагмент одного из файлов, полученных из собранных данных одного из пользователя, представляющий набор характеристических признаков этого пользователя, представлен в таблице 5:

Таблица 5. Пример таблицы с преобразованными данными

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| keycode | HD | PPD | RPD | RRD |
| 66 | 0.0477297 | 0.5526373 | 0.5049076 | 0.5814094 |
| … | … | … | … | … |

При анализе преобразованного файла можно заметить, что некоторые значения параметра «Время перескока между клавишами» имеют отрицательное значение. Эти значения связаны с наложением движения – одновременным движением нескольких пальцев у пользователя. Наложение нажатий клавиш происходит, когда одна кнопка еще не отпущена, но при это другая уже нажимается. Количество наложений зависит от скорости печати. В основном они происходят, когда нажимаются соседние клавиши разными пальцами.

## 3.3 Шкалирование данных

Современные методы машинного обучения, алгоритмы которых основаны на вычислении расстояния между точками в пространстве, чувствительны к шкалированию данных. Поэтому выборки с различными диапазонами в признаках могут спровоцировать искаженное восприятие данных моделью. Для решения этой проблемы существует две основные техники: нормализация и стандартизация. Благодаря этому, ни один из признаков выборки не доминирует при обучении модели, и классификатор корректно учтет каждый из них.

Нормализацией данных называют масштабирование данных таким образом, что выходные данные находятся в диапазоне [0, 1], сохраняя при этом отношения между ними. Каждый элемент признака вычисляется по формуле (1):

(1),

где – входной элемент признака;

– наибольший элемент признака;

– наименьший элемент признака;

– выходной элемент.

При стандартизации каждый признак данных после преобразования имеет среднее значение, равное нулю, и стандартное отклонение, равное единицы, что делает влияние каждого признака на модель в одинаковой степени. Вычисляется каждый элемент в этом случае по формуле (2):

(2),

где – выходной элемент;

– входной элемент;

– среднее арифметическое;

– стандартное отклонение.

Стандартизация применяется в случаях, когда данные соответствуют гауссовскому распределению. Построим гистограммы времени удержания клавиш нескольких пользователей. Результаты представлены на рисунках 5, 6, 7:

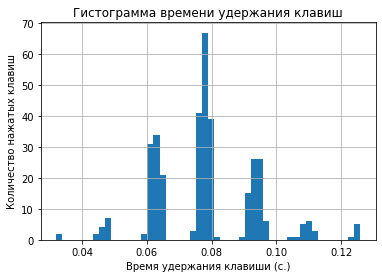


Рисунок 5. Гистограмма времени удержания клавиш тестового пользователя

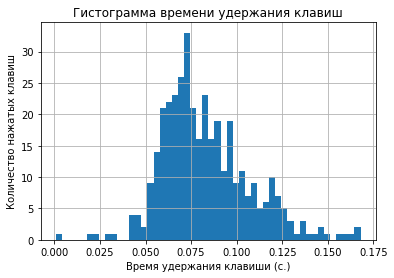


Рисунок 6. Гистограмма времени удержания клавиш тестового пользователя

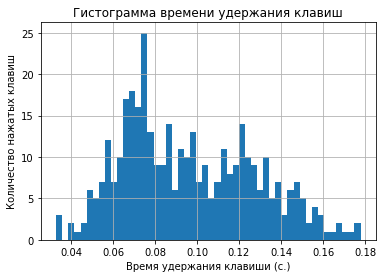


Рисунок 7. Гистограмма времени удержания клавиш тестового пользователя

По графикам видно, что распределения так или иначе являются нормальными (гауссовскими) или бимодальными (объединение двух нормальных распределений), поэтому стандартизация в данном случае лучше применима, что также показали проводимые тесты.

## 3.4 Вывод

Таким образом, исходя из представленных данных можно сделать вывод о механизме снятия данных, а также его анализа. Данные показатели позволят обучить модель для определения легитимности сотрудника на рабочем месте. Параметры могут сниматься непрерывно с клавиатуры и операция снятия данных с клавиатуры может осуществляться на фоне работы сотрудника и не влиять на его деятельность за рабочим местом, что также влияет на комфорт работы самого сотрудника, а также автоматизирует аутентификацию сотрудника, при которой аутентифкатором является его работа на клавиатуре.

Из проведенного обзора собранных данных и полученных из них характеристических признаков можно сделать следующие выводы:

1. Для анализа поведения пользователя по клавиатурному почерку записываются следующие данные: код клавиши, тип события, и время произошедшего события. При смене клавиатуры необходимо заново составлять обучающую выборку, так как устройства разных производителей имеют различные технические характеристики и расположение клавиш, что понижает качество распознавания.
2. После сбора данных для обучения модели необходимо преобразовать их в набор характеристических признаков. Рассматриваются такие признаки, как время удержания клавиши, а также время переноса между клавишами.
3. Для масштабирования признаков было решено применять нормализацию, так как эта техника лучше подходит для имеющихся данных. Она позволяет уровнять влияние каждого признака на модель искусственного интеллекта.

# 4. Обоснование и выбор алгоритмов для построения моделей машинного обучения и разработка системы

## 4.1 Обзор применимых алгоритмов машинного обучения и выбор подходящего

Следующим этапом в создании СЗИ является выбор алгоритма модели машинного обучения. Так как для обучения искусственного интеллекта доступны данные лишь легитимного пользователя, большинство алгоритмов оказываются неприменимыми для решения данной задачи. Поэтому рассматривались методы одноклассовой классификации, задача которых классифицировать новые данные как похожие или отличные от обучающего набора. Наиболее популярные алгоритмы: одноклассовый метод k-ближайших соседей, модель гауссовых смесей и одноклассовый метод опорных векторов. Все перечисленные методы являются геометрическими и основаны на расчётах различных мер близости. Рассмотрим суть каждого из них:

1. Одноклассовый метод k-ближайших соседей. Одноклассовый метод k-ближайших соседей. Наиболее простой из трех алгоритмов. Метод основан на анализе расстояний между объектами. Сначала высчитывается расстояние между тестовым объектом и ближайшим к нему объекту обучающей выборки. Затем высчитывается расстояние между этим объектом обучающей выборки и его ближайшим соседом. Если отношение первого расстояния ко второму меньше, чем параметр , то объект считается легитимным. Данный параметр подбирается экспериментально.
2. Модель гауссовых смесей. Как и было отмечено в пункте 3.3, данные динамики работы пользователей с клавиатурой соответствуют нормальному или бимодальному распределению, однако у каждого пользователя параметры этих распределений отличаются. Поэтому можно сравнивать распределение тестируемого набора данных с обучающей выборкой и делать прогноз на основе сравнения. Недостаток данного способа в сложности реализации и трудных вычислениях.
3. Одноклассовый метод опорных векторов. В классическом случае в задаче бинарной классификации суть метода опорных векторов заключается в следующем: разделить признаки, относящиеся к разным классам, с помощью плоскости так, чтобы расстояние до ближайшей точки каждого класса было максимальным. В случае одноклассовой классификации разделяющая гиперплоскость проводится так, чтобы значения в обучающей выборки находились как можно дальше от начала координат. Тогда по одну сторону гиперплоскости будут находиться легитимные значения, относящиеся к тренировочным данным, а по другую – аномальные значения или выборы. Визуально данные метод представлен на рисунке 8:

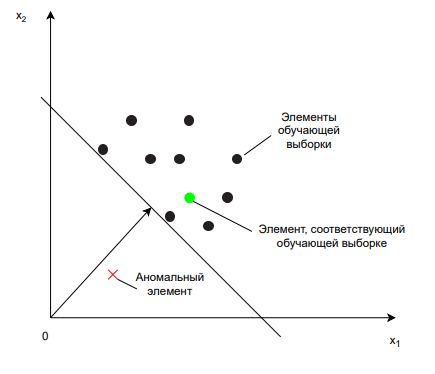


Рисунок 8. Визуальное описание работы одноклассового метода опорных векторов

По исследованиям именно последний метод наиболее часто применяется для распознавания пользователя по динамике нажатия клавиш и стабильно показывает высокую точность [[5](#_Список_литературы)]. Также, его достоинством является использование небольшого количества данных для обучения. Следовательно, он был выбран в качестве алгоритма модели машинного обучения.

## 4.2 Подбор параметров модели машинного обучения на основе собранных данных

Качество работы выбранного алгоритма одноклассового метода опорных векторов зависит от таких параметров как:

* nu – верхняя граница доли ошибок при обучении
* gamma – коэффициент функции ядра, которая имеет вид, представленный в формуле (3):

(3),

где – значения до отображения;

– значения после отображения.

Функцией ядра называется такая функция, которая переносит все входные данные в бо́льшую размерность. Это необходимо, так как не всегда данные линейно разделимы, то есть не всегда существует такая гиперплоскость, которой можно разделить значения. После повышения размерности данные становятся линейно разделимыми.

Для подбора этих параметров проводился эксперимент. Сначала пользователем, на данных которого модель обучалась, был введен текст, отличающийся от текста для обучения. После этого итеративно модель обучалась с разными параметрами в заданном диапазоне. Диапазон для обучение следующий:

* nu от 0,0025 до 1 с шагом 0,0025,
* gamma от 0,01 до 5 с шагом 0,01

Модель прогнозировала результат для 4-ех блоков характеристических признаков разных нелегитимных пользователей и 2-ух различных блоков собранных данных легитимного пользователя. Высчитывалась вероятность принадлежности данных к обучающей выборке, высчитывалось среднее значение для каждой из групп и находила разность между ними. Наилучшей считалась модель с наибольшей разностью, так как это характеризует тот факт, что модель наилучшим образом умеет отличать пользователя от злоумышленника. Полный код программы для эксперимента находится в Приложении 3.

Результаты экспериментов показали, что по такой оценке модель имеет наилучший результат с параметрами, представленными на рисунке 9:



Рисунок 9. Вывод наилучших параметров при проведенном эксперименте

Рассмотрим предсказанные значения для модели с этими параметрами. Сначала оценим выдаваемые результаты для двух тестовых выборок легитимного пользователя. Оценка представлена на рисунке 10:



Рисунок 10. Вероятности для легитимного пользователя

Из теста видно, что по блоку в пятнадцать символов модель на двух разных тестовых выборках спрогнозировала вероятности для легитимного пользователя в 0,8 и 0,93. Среднее значение составляет 0,8666(6)

Результаты прогнозирования вероятностей для четырех различных блоков характеристических признаков пользователей, имитирующих действия злоумышленника, представлены на рисунке 11:

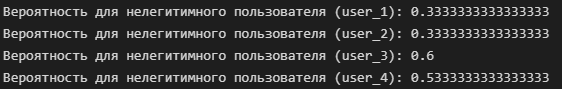


Рисунок . Вероятности для нелегитимных пользователей

Среднее значение полученных данных составляет 0,45. Исходя из этих данных можно установить порог, ниже которого модель будет считать пользователя неправомерным, равным 0,7.

## 4.3 Описание структуры разрабатываемого СЗИ

Разрабатываемое программное средство имеет клиент-серверную архитектуру. Структура взаимодействия модулей клиента и сервера представлена на рисунке 12:

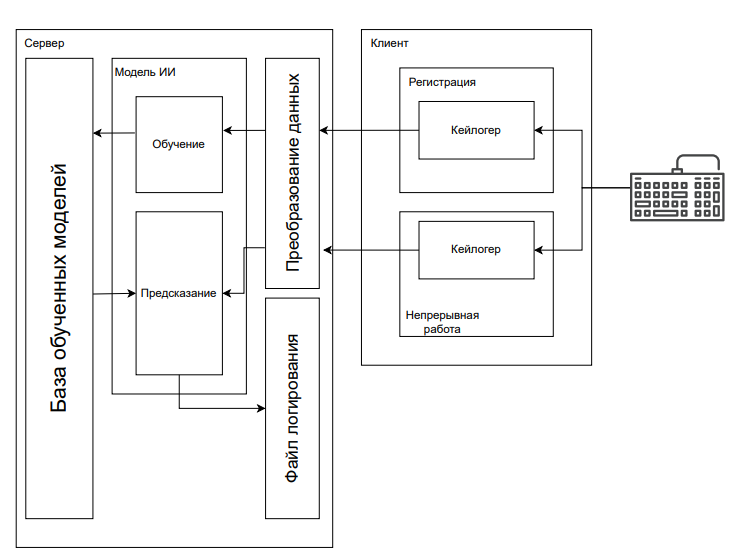


Рисунок . Структурная схема разрабатываемого прототипа

### 4.3.1 Описание структуры и алгоритм работы сервера

Программный комплекс сервера включает в себя:

* model.py - модуль с моделью искусственного интеллекта для обучения и проверки пользователя, позволяющий сохранять обученную модель в специальный файл. Полный код расположен в Приложении 4;
* keys\_processed.py - разработанный в пункте 3.2 модуль для преобразования входных данных в набор характеристических функций. Полный код расположен в Приложении 2;
* app.py – веб-сервер, принимающий запросы от клиента и отправляющий результат анализа модели обратно клиенту. Полный код расположен в Приложении 5;
* файловая структура для сохранения собранных данных, преобразованных данных, а также обученных моделей для последующей проверки пользователей.

Для реализации серверной части использованы следующие технологии:

* Python (как основной язык программирования)
* Flask (веб-фреймворк)
* Scikit-learn (библиотека для машинного обучения)
* Pandas (библиотека для анализа данных)

Работа с моделью машинного обучения реализована с помощью двух методов:

* create\_and\_train\_model(user) – метод, предназначенный для создания модели, ее обучения и сохранения модели в файл для последующего использования;
* check\_user(user, data) – метод, предназначенный для анализа моделью входных данных и предсказания вероятности принадлежности этих данных пользователю.

Веб-сервер реализован по архитектуре REST. REST – это архитектурный стиль создания веб-приложений, описывающий взаимодействие компонентов в сети. Используется для реализации связи между клиентом и сервером. В данном случае эта связь происходит с помощью API – программный интерфейс приложения – представляющий собой специальные URI адреса. Список доступных URI веб-сервера представлен в таблице 6:

Таблица . Список URI веб-сервера

|  |  |  |
| --- | --- | --- |
| URI | Метод | Описание |
| <url-адрес>/new\_client | GET | Возвращает описание назначения URI |
| POST | Регистрирует на сервере нового клиента |
| <url-адрес>/is\_registered | GET | Возвращает описание назначения URI |
| POST | Проверяет, существует ли пользователь на сервере |
| <url-адрес>/check\_client | GET | Возвращает описание назначения URI |
| POST | Аутентификация пользователя в процессе работы |

### 4.3.2 Описание структуры и алгоритм работы клиента

Клиент состоит из двух модулей и одной папки:

* client.py - основная программа, отвечающая за передачу данных на сервер и прием данных. Полный код расположен в Приложении 6;
* keyloger.py - преобразованный модуль из пункта 3.1, позволяющий собирать события клавиатуры. Полный код расположен в Приложении 7.
* Папка с файлами, содержащими в себе тексты для обучения.

Для реализации клиента использованы следующие технологии:

* Python (как основной язык программирования)
* Requests (библиотека для автоматизации http-запросов)
* Pynput (библиотека для считывания событий клавиатуры)

На рисунке 13 показан алгоритм работы клиента:

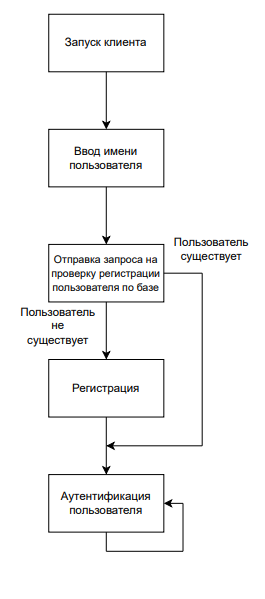


Рисунок . Алгоритм работы клиента

При запуске клиента дополнительным аргументом указывается IP-адрес сервера. Для проверки прохождения регистрации клиент просит ввести его имя, после чего происходит запрос на сервер. Если пользователь ранее проходил процесс обучения и его модель на сервере существует, процесс регистрации пропускается и начинается аутентификация пользователя. Иначе клиент просит ввести два небольших текста, чтобы считать параметры пользователя и отправить их на сервер для дальнейшего обучения модели. Эти тексты выбираются случайным образом из набора, хранящегося в папке клиента.

После получения от сервера информации об удачном обучения клиент непрерывно начинает считывать поток событий клавиатуры. Для отправки данных на сервер необходимо разбивать этот поток на окна. Окном называется последовательный фрагмент потока событий. Размер этого окна имеет достаточно большую важность, так как, с одной стороны, окна большего размера передают больше информации для проверки, но при этом дает злоумышленнику больше возможностей до выявления вторжения. Для получения этих окон можно разбивать потоки по двум принципам:

1) По количеству событий. По исследованиям - предпочтительно использовать при средней клавиатурной активности пользователя в процессе работы.

2) По длительности работы. Предпочтительно использовать при высокой активности на протяжении всего времени работы за компьютером.

В обоих случаях длина окна может быть постоянной, либо находиться в заданном интервале. Также можно отправлять данные на сервер при длительной паузе в работе, даже если заданная длина не была достигнута.

Для разрабатываемой СЗИ было выбрано окно в 15-16 клавиш (в зависимости от количества наложений, чтобы последним действием в перечне событий окна всегда было отжатие клавиши), потому что это значение лежит в интервале копирования. Интервал копирования – это число символов, которые могут быть напечатаны после однократного просмотра текста. Установлено, что это значение в обычной ситуации перепечатывания текста у опытного пользователя составляет в среднем 14,6 символов [[6](#_Список_литературы)].

На сервер данные отправляются в формате JSON. Структура одного отправляемого пакета данных представлена на рисунке 14:

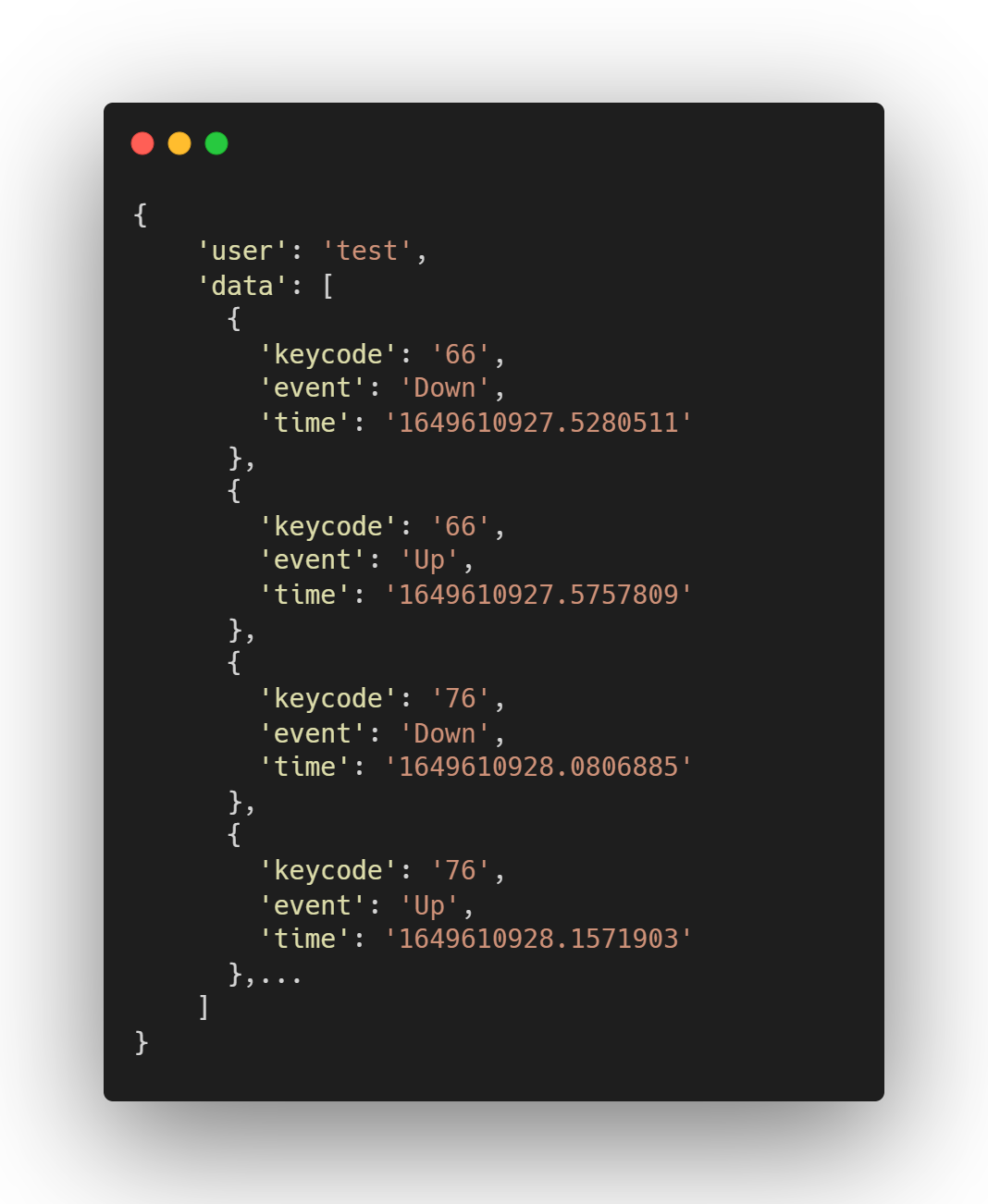


Рисунок . Пример JSON-объекта, отправляемого клиентом

## 4.4 Вывод

Таким образом, можно сделать вывод, для выбора определенного алгоритма машинного обучения были изучены самые популярные методы, которые могут быть применимы для данных клавиатурного почерка. Выбор был сделан в пользу одноклассового метода опорных векторов, так как, учитывая открытые источники, он чаще всего используется для обучения, где входными данными являются динамика нажатия клавиш.

Были рассмотрены параметры модели, от которых зависит работа алгоритма, и после ряда тестов была выбрана граница, по которой происходит разделение пользователя на легитимного и злоумышленника.

Выбрав алгоритм машинного обучения, была разработана структура клиент-серверного приложения. Суть клиентского приложения состоит в считывании данных с клавиатуры пользователя и отправке их на сервер. Данные, считанные с клавиатуры, передаются на сервер в виде JSON объекта. Данный формат был выбран из соображений совместимости с используемым фреймворком и языком, что позволяет не создавать отдельные классы-конверторы для корректной работы программы.

Клиентское приложение для получения данных для эталонной модели и обучения при первом запуске просит пройти пользователя этап регистрации. После сбора данных приложение продолжает свою работу в фоновом режиме, не мешая сотруднику выполнять свои обязанности.

# 5. Тестирование работы реализованной программы по выявлению НСД и оценивание полученных результатов

## 5.1 Пользовательское тестирование программы

Для тестирования системы сначала происходит запуск сервера, так как работа клиента, а, следовательно, и проведение аутентификации без него невозможно.

В тестовой среде веб-сервер поднять в режиме разработки для удобной отладки системы и более информативного вывода ошибок при неправильной работе или ошибках при связи с клиентом. Сервер поднимается по адресу 127.0.0.1 и доступ к нему запущенные клиенты могут получить только в пределах локального компьютера. Результат запуска сервера представлен на рисунке 15:

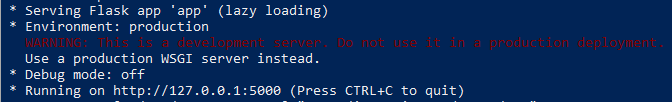


Рисунок . Вывод при запуске тестового сервера

При использовании разработки в рабочих условиях необходимо дополнительно реализовать WSGI – интерфейс, позволяющий связать python-приложение с «боевым» сервером, таким как NGINX.

После запуска сервера происходит запуск клиента. Работа с клиентом происходит в командной строке. Сначала высвечивается сообщение с просьбой о вводе имени. Приложение просит ввести два текста различного содержания и выводит уведомление об удачной регистрации. На рисунке 16 показан процесс первичного запуска клиента и его регистрация:

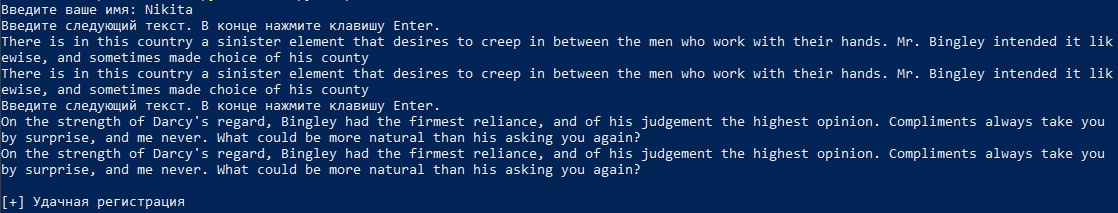


Рисунок . Процесс регистрации на клиенте

При повторном запуске клиента с таким же именем высвечивается уведомление, о том, что модель для данного пользователя уже обучена и процесс аутентификации начат, как видно на рисунке 17:



Рисунок . Вывод при вводе существующего пользователя

При это на сервере в режиме отладки появляются сообщения о приходящих с клиента запросах, что показано на рисунке 18:



Рисунок . Вывод запросов от клиента на сервере

Все результаты оценки модели сохраняются в файл логирования на сервере. Фрагменты этого файла представлены на рисунках 19, 20:

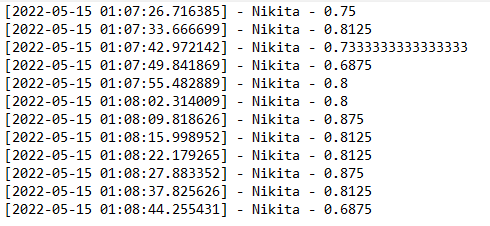


Рисунок . Фрагмент журнала событий клиента

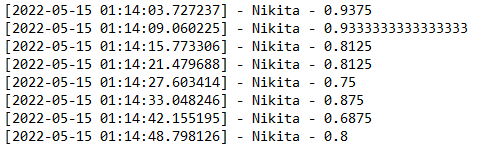


Рисунок . Фрагмент журнала событий клиента

При этом, если одного из пользователей система посчитала нелегитимным, уведомление об этом также появляется на экране с указанием имени, от которого был запущен клиент (см. рисунок 21):



Рисунок . Уведомление на стороне сервера при оценке модели пользователя как злоумышленника

Когда клавиатурой начал пользоваться другой пользователь, система стала намного чаще выдавать негативные результаты, что видно из фрагмента логов на рисунке 22:

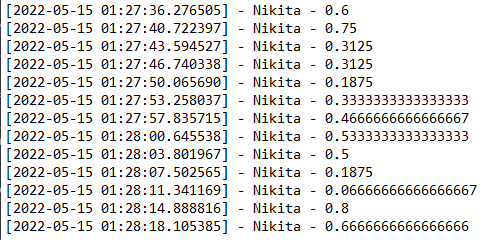


Рисунок . Журнал событий сервера при смене пользователя

## 5.2 Оценивание полученных результатов

Исследуя полученные результаты, можно сказать, что, в случае валидного пользователя, программа, в большинстве случаев, выдает результат присутствия легитимного оператора на рабочем месте. В представленных результатах только в 3 случаях из 20 программа неверно определяла принадлежность пользователя, что означает, в 85 процентах случаев приложение верно определяет легитимного пользователя.

Протестировав работу программы на нелегитимном пользователе, были получены обратные результаты. Из 13 процессов аутентификации только 2 из них система определила нелегитимного пользователя как валидного, что также означает, что в 84,6 процентах случаев система верно определит злоумышленника на рабочем месте сотрудника.

Стоит уделить также внимание и недостаткам данного приложения. Клавиатурный почерк является динамической характеристикой поведения человека, что означает влияние внешних факторов, давящих на сотрудников, способных повлиять на стиль набора символов на клавиатуре. Наблюдается сильная зависимость от психофизического состояния оператора. Если человек заболел, то он вполне, вероятно, не сможет аутентифицироваться. Поэтому применение данного приложения в качестве инструмента непрерывной аутентификации актуально для стабильного состояния сотрудников. Также возможно обновлять эталонную модель в процессе работы. Таким образом, возможно достижение непрерывной аутентификации сотрудника в зависимости от различных его состояний.

## 5.3 Вывод

Исследования работы приложения показывают высокую точность в определении легитимности пользователя, который находится на рабочем месте. Такая точность была достигнута благодаря качественно выбранным параметрам модели.

Принцип работы клиентского приложения заключается в прохождении регистрации, вводе имени, затем создания данных для эталонной модели для обучения. Данный этап представляется в приложении в виде двух последовательных наборов текстов различного содержания. Тексты подобраны в приложении таким образом, чтобы в них содержалось как можно больше букв из алфавита, тем самым делая более качественную эталонную модель.

Исследование показало, что на набор текста для эталонной модели, а также последующие операции аутентифицирования пользователя, очень сильное влияет оказывает психофизическое состояние человека. При различных обстоятельствах сотрудника наблюдается различные результаты прохождения аутентификации. Сильное влияние оказывает утомленность и раздраженность сотрудника. Под влиянием данных чувств человек замедляет свой набор текста, либо, наоборот, ускоряет его. Отследить данное состояние с помощью приложения не удастся и аутентификация пользователя будет провалена. Но приложение позволяет в течение работы изменять данные эталонной модели, проходя заново этап регистрации. Это позволяет качественно реагировать на изменение состояния человека и стиля клавиатурного набора символов. Приложение показывает высокую точность определения валидности пользователя в системе.

# Заключение

Основной задачей данной работы была разработка прототипа системы защиты информации от несанкционированного доступа для непрерывной аутентификации пользователя в веб-среде на основе его поведения.

Для этого на начальной стадии были рассмотрен биометрические статические и динамические характеристики, проведено их сравнение и выбран наиболее подходящий вариант для непрерывной аутентификации – динамика нажатий клавиш. Также были рассмотрены методы, основанные на поиске шаблонов при использовании сети Интернет. Могут анализироваться такие действия пользователя, как его перемещение по страницам веб-ресурсов, так и его взаимодействие с каждым в отдельности.

Для обзора подходов по созданию таких систем были рассмотрены существующие на рынке решения. Некоторые из них работают по архитектуре клиент-сервер, при которой данные с клавиатуры отправляются на главный сервер, на котором и происходит оценивание характеристик пользователя, а некоторые полностью проводят аутентификацию на локальном компьютере.

Далее был описан процесс сбора событий о нажатиях кнопок клавиатуры и реализована программа для выполнения этой задачи. Впоследствии, эти данные были преобразованы в набор характеристик и было произведено масштабирование этих признаков, чтоб передать эти данные машине для обучения.

В дальнейшем были рассмотрены применяемые алгоритмы и выбран наиболее популярные из них, показывающий наилучший результат в решении задач по аутентификации по клавиатурному почерку. Для нахождения наилучших параметров модели был проведен эксперимент, заключающийся в подборе таких параметров, которые бы лучшим образом отделяли легитимного пользователя от остальных. Разработана структура, реализован и опробован образец программного обеспечения, использующий выбранный алгоритм для обнаружения аномального поведения пользователей по особенностям работы с клавиатурой компьютера.

Применение полученного средства показало высокий процент идентификации пользователя, при этом тесты проводились как для верного оператора, так и для имитирующего злоумышленника.

Разработанная система в будущем может быть улучшена как функционально, так и визуально с помощью:

1. Добавления дополнительных поведенческих характеристик анализа клавиатурного почерка. Можно вычислять такие параметры, как скорость набора текста, а также статистические значения входящих параметров. Также можно анализировать комбинации 2-ух, 3-ех и более нажатых клавиш.
2. Добавления дополнительных биометрических и поведенческих признаков, что позволит идентифицировать пользователя параллельно с помощью нескольких признаков и на основе комплексного анализа выявлять злоумышленника.
3. Подбора параметры модели отдельно для каждого пользователя.
4. Добавления дополнительного алгоритма машинного обучения или нейронной сети для более точного результата.
5. Добавления графического интерфейса для клиента, а также веб-интерфейса на сервере для администратора.
6. Использования шифрованного соединения при передаче данных между клиентом и пользователем.

Реализация этих улучшений позволит создать действительно надежную систему аутентификации из разработанного прототипа. Однако, как показывает практика, любое средство защиты информации лучше применять в комплексе для более надежной защиты и точных результатов работы, и данное средство не является исключениям.

# Список литературы

1. В.А. Ворона, В.А. Тихонов. Системы контроля и управления доступом. – Москва.: Издательство «Горячая линия – Телеком», 2010. (Дата обращения 20.04.2022)
2. Биометрическая идентификация [электронный ресурс] URL: <http://www.techportal.ru/glossary/biometricheskaya_identifikaciya.html> (Дата обращение 20.04.202)
3. Myriam Abramson, David W. Aha. User Authentication from Web Browsing Behavior [текст] // The Twenty-Sixth International FLAIRS Conference, - 2013 (Дата обращения 22.04.2022)
4. Эксперимент в Яндексе. Как идентифицировать взломщика с помощью машинного обучения [электронный ресурс] URL: <https://habr.com/ru/company/yandex/blog/230583/> (Дата обращения 23.04.2022)
5. Killourhy K. S., Maxion R. A. Comparing anomaly-detection algorithms for keystroke dynamics //Dependable Systems & Networks, 2009. – IEEE, 2009. – С. 125-134. (Дата обращения 05.05.2022)
6. Timothy A. Salthouse. Perceptual, Cognitive, and Motoric Aspects of Transcription Typing // Psyhol. Bull. -1999. P. 303-319. (Дата обращения 11.05.2022)

# Приложение 1

from pynput import keyboard

import time

class KeyLogger():

    def \_\_init\_\_(self, name, filename: str = "keylogs.csv") -> None:

        self.name = name

        self.filename = filename

        with open(self.filename, 'w') as logs:

            logs.writelines('keycode,event,time\n')

    def get\_key\_id(self, key):

        try:

            return key.vk

        except AttributeError:

            return key.value.vk

    def on\_press(self, key):

        if key not in (keyboard.Key.esc, keyboard.Key.shift\_l, keyboard.Key.alt\_l, keyboard.Key.ctrl\_l, keyboard.Key.enter, keyboard.Key.delete):

            with open(self.filename, 'a') as logs:

                logs.writelines('{},{},{}\n'.format(self.get\_key\_id(key), 'Down', time.time()))

    def on\_release(self, key):

        if key not in (keyboard.Key.esc, keyboard.Key.shift\_l, keyboard.Key.alt\_l, keyboard.Key.ctrl\_l, keyboard.Key.enter, keyboard.Key.delete):

            with open(self.filename, 'a') as logs:

                logs.writelines('{},{},{},{}\n'.format(self.get\_key\_id(key), 'Up', time.time()))

    def main(self):

        listener = keyboard.Listener(on\_press=self.on\_press, on\_release=self.on\_release)

        listener.start()

        try:

            listener.wait()

            input()

        finally:

            listener.stop()

if \_\_name\_\_ == '\_\_main\_\_':

    name = input('Введите ваше имя: ')

    logger = KeyLogger(name, name+'\_logs.csv')

    print('Введите следующий текст. В конце нажмите клавишу Enter.')

    print('When the ladies removed after dinner, Elizabeth ran up to her sister, and seeing her well guarded from cold. What could be more natural than his asking you again?')

    logger.main()

    print('Теперь введите. В конце нажмите клавишу Enter.')

    print("These reflections just here are occasioned by the circumstance that after we were all seated at the table. But to be candid without ostentation or design to take the good of everybody's character and make it still better.")

    logger.main()

# Приложение 2

import pandas as pd

import warnings

warnings.filterwarnings('ignore')

def convert\_json(data):

    return pd.json\_normalize(data)

def save\_capture\_data\_to\_file(user, data):

    data.to\_csv(f'keystroke/userdata/user\_logs/{user}\_logs.csv',  index=False)

def read\_capture\_data\_from\_file(user):

    return pd.read\_csv(f'keystroke/userdata/user\_logs/{user}\_logs.csv')

def save\_processed\_data\_to\_file(user, data):

    data.to\_csv(f'keystroke/userdata/processed\_data/{user}\_processed\_data.csv', index=False)

def processing\_keys(data):

    df = pd.DataFrame(columns=['keycode','HD','PPD','RPD', 'RRD'])

    event\_len = len(data)

    for i in range(event\_len-5):

        if str(data.iloc[i].event) == 'Down':

            finalData = {}

            cur\_key\_press\_time = data.iloc[i].time

            cur\_key\_release\_time =  data[(data.keycode==data.iloc[i].keycode)&(data.event=='Up')&(data.time>data.iloc[i].time)].iloc[0].time

            next\_key\_press\_time = data[(data.event=='Down')&(data.time>data.iloc[i].time)].iloc[0].time

            next\_key\_keycode = data[(data.event=='Down')&(data.time>data.iloc[i].time)].iloc[0].keycode

            next\_key\_release\_time =  data[(data.keycode==next\_key\_keycode)&(data.event=='Up')&(data.time>next\_key\_press\_time)].iloc[0].time

            finalData['keycode'] = data.iloc[i].keycode

            finalData['HD'] = cur\_key\_release\_time-cur\_key\_press\_time

            finalData['PPD'] = next\_key\_press\_time-cur\_key\_press\_time

            finalData['RPD'] = next\_key\_press\_time-cur\_key\_release\_time

            finalData['RRD'] = next\_key\_release\_time-cur\_key\_release\_time

            df = df.append(finalData,ignore\_index=True)

    return df

# Приложение 3

from sklearn import preprocessing

from sklearn.svm import OneClassSVM

import numpy as np

import pandas as pd

import warnings

warnings.filterwarnings('ignore')

test\_data=pd.read\_csv('server/keystroke/Nikita\_test\_logs.csv')

df = pd.DataFrame(columns=['keycode','HD','PPD','RPD', 'RRD'])

user = 'Nikita'

event\_len = len(test\_data)

for i in range(event\_len-3):

    if str(test\_data.iloc[i].event) == 'Down':

        finalData = {}

        cur\_key\_press\_time = test\_data.iloc[i].time

        cur\_key\_release\_time =  test\_data[(test\_data.keycode==test\_data.iloc[i].keycode)&(test\_data.event=='Up')&(test\_data.time>test\_data.iloc[i].time)].iloc[0].time

        next\_key\_press\_time = test\_data[(test\_data.event=='Down')&(test\_data.time>test\_data.iloc[i].time)].iloc[0].time

        next\_key\_keycode = test\_data[(test\_data.event=='Down')&(test\_data.time>test\_data.iloc[i].time)].iloc[0].keycode

        next\_key\_release\_time =  test\_data[(test\_data.keycode==next\_key\_keycode)&(test\_data.event=='Up')&(test\_data.time>next\_key\_press\_time)].iloc[0].time

        finalData['keycode'] = test\_data.iloc[i].keycode

        finalData['HD'] = cur\_key\_release\_time-cur\_key\_press\_time

        finalData['PPD'] = next\_key\_press\_time-cur\_key\_press\_time

        finalData['RPD'] = next\_key\_press\_time-cur\_key\_release\_time

        finalData['RRD'] = next\_key\_release\_time-cur\_key\_release\_time

        df = df.append(finalData,ignore\_index=True)

df.keycode = df.keycode.apply(int).apply(str)

n\_scaler = preprocessing.StandardScaler()

data = pd.read\_csv('server/keystroke/userdata/processed\_data/Nikita\_processed\_data.csv')

col\_names = ['HD', 'PPD', 'RPD', 'RRD']

data\_s = n\_scaler.fit\_transform(data[col\_names])

data\_s = pd.DataFrame(data\_s, columns = col\_names)

data\_s['keycode']=data.keycode.astype('str')

nu = np.arange(0.0025, 1, 0.0025)

gamma = np.arange(0.05, 5, 0.05)

max = 0

for n in nu:

    for g in gamma:

        clf = OneClassSVM(nu=n, kernel="rbf", gamma=g)

        clf.fit(data\_s)

        user1\_data = pd.read\_csv('server/keystroke/userdata/processed\_data/user1\_processed\_data.csv')[:15]

        user1\_data\_s = n\_scaler.transform(user1\_data[col\_names])

        user1\_data\_s = pd.DataFrame(user1\_data\_s, columns = col\_names)

        user1\_data\_s['keycode']=user1\_data.keycode.values.astype('str')

        \_, user1\_count =np.unique(clf.predict(user1\_data\_s), return\_counts=True)

        user2\_data = pd.read\_csv('server/keystroke/userdata/processed\_data/user2\_processed\_data.csv')[:15]

        user2\_data\_s = n\_scaler.transform(user2\_data[col\_names])

        user2\_data\_s = pd.DataFrame(user2\_data\_s, columns = col\_names)

        user2\_data\_s['keycode']=user2\_data.keycode.values.astype('str')

        \_, user2\_count =np.unique(clf.predict(Duser2\_data\_s), return\_counts=True)

        user3\_data = pd.read\_csv('server/keystroke/userdata/processed\_data/user3\_processed\_data.csv')[:15]

        user3\_data\_s = n\_scaler.transform(user3\_data[col\_names])

        user3\_data\_s = pd.DataFrame(user3\_data\_s, columns = col\_names)

        user3\_data\_s['keycode']=user3\_data.keycode.values.astype('str')

        \_, user3\_count = np.unique(clf.predict(user3\_data\_s), return\_counts=True)

        user4\_data = pd.read\_csv('server/keystroke/userdata/processed\_data/user4\_processed\_data.csv')[:15]

        user4\_data\_s = n\_scaler.transform(user4\_data[col\_names])

        user4\_data\_s = pd.DataFrame(user4\_data\_s, columns = col\_names)

        user4\_data\_s['keycode']=user4\_data.keycode.values.astype('str')

        \_, user4\_count = np.unique(clf.predict(user4\_data\_s), return\_counts=True)

        df\_s = n\_scaler.transform(df[col\_names])

        df\_s = pd.DataFrame(df\_s, columns = col\_names)

        df\_s['keycode']=df.keycode.astype('str')

        \_, test\_valid\_counts = np.unique(clf.predict(df\_s[20:35]), return\_counts=True)

        \_, test\_valid\_counts\_2 = np.unique(clf.predict(df\_s[55:70]), return\_counts=True)

        try:

            false\_rate = (user1\_count[1]+user2\_count[1]+user3\_count[1]+user4\_count[1])/60

            true\_rate = (test\_valid\_counts[1]+test\_valid\_counts\_2[1])/30

            dif\_len\_rate=(true\_rate-false\_rate)

            if dif\_len\_rate>max:

                max\_i = i

                max = dif\_len\_rate

                max\_n = n

                max\_g = g

        except:

            pass

print(f'nu = {max\_n}, gamma = {max\_g}, max\_difference = {max}')

# Приложение 4

import joblib

import numpy as np

import pandas as pd

from sklearn.preprocessing  import StandardScaler

from sklearn.svm import OneClassSVM

def create\_and\_train\_model(user):

    scaler = StandardScaler()

    data = pd.read\_csv(f'keystroke/userdata/processed\_data/{user}\_processed\_data.csv')

    col\_names = ['HD', 'PPD', 'RPD', 'RRD']

    data\_s = scaler.fit\_transform(data[col\_names])

    data\_s = pd.DataFrame(data\_s, columns = col\_names)

    data\_s['keycode']=data.keycode.values.astype('str')

    clf = OneClassSVM(nu = 0.0025, gamma = 0.2, kernel="rbf")

    clf.fit(data\_s)

    filename\_norm = f'keystroke/models/{user}\_norm.sav'

    filename\_model = f'keystroke/models/{user}\_model.sav'

    joblib.dump(scaler, filename\_norm)

    joblib.dump(clf, filename\_model)

def check\_user(user, data):

    filename\_scaler = f'keystroke/models/{user}\_norm.sav'

    scaler = joblib.load(filename\_scaler)

    col\_names = ['HD', 'PPD', 'RPD', 'RRD']

    data\_s = scaler.transform(data[col\_names])

    data\_s = pd.DataFrame(data\_s, columns = col\_names)

    data\_s['keycode']=data.keycode.values.astype('str')

    filename\_model = f'keystroke/models/{user}\_model.sav'

    loaded\_model = joblib.load(filename\_model)

    pred = loaded\_model.predict(data\_s[:15])

    cls, counts = np.unique(pred, return\_counts=True)

    try:

        return counts[1]/len(data)

    except:

        if cls[0]==-1:

            return 0

        else:

            return 1

# Приложение 5

from flask import Flask, request, jsonify

import datetime

from keystroke.keys\_processed import \*

from keystroke.model import \*

app = Flask(\_\_name\_\_)

def find\_user(user):

    with open('users.txt', 'r') as f:

        users = f.read().splitlines()

    return True if user in users else False

def add\_user(user):

    with open('users.txt', 'a+') as f:

        f.write(f'{user}\n')

def write\_to\_log\_file(user, proba):

    with open('server\_logs.txt', 'a+') as f:

        f.write(f'[{datetime.datetime.now()}] - {user} - {proba}\n')

@app.route("/new\_client",  methods=['GET', 'POST'])

def new\_client():

    if request.method == 'POST':

        data = request.get\_json()

        try:

            user = data['user']

            add\_user(user)

            keys = data['data']

            keys = convert\_json(keys)

            save\_capture\_data\_to\_file(user, keys)

            processed\_keys = processing\_keys(keys)

            save\_processed\_data\_to\_file(user, processed\_keys)

            create\_and\_train\_model(user)

            return jsonify({'user':user, 'status':'successfully'})

        except:

            return jsonify({'user':user, 'status':'unsuccessfully'})

    else:

        return jsonify({'description':'Add new user to system'})

@app.route("/is\_registered",  methods=['GET', 'POST'])

def is\_registred():

    if request.method == 'POST':

        data = request.get\_json()

        user = data['user']

        if find\_user(user):

            return jsonify({'user':user, 'status':'is\_exist'})

        else:

            return jsonify({'user':user, 'status':'does\_not\_exist'})

    else:

        return jsonify({'description':'Check if user is registered'})

@app.route("/check\_client",  methods=['GET', 'POST'])

def check\_client():

    if request.method == 'POST':

        data = request.get\_json()

        user = data['user']

        keys = data['data']

        keys = convert\_json(keys)

        processed\_keys = processing\_keys(keys)

        proba = check\_user(user, processed\_keys)

        write\_to\_log\_file(user, proba)

        if  proba>0.7:

            return jsonify({'user':user, 'result': f'valid - {proba}'})

        else:

            print(f'[!!!] {user} - {proba}')

            return jsonify({'user':user, 'result': f'intruder - {proba}'})

    else:

        return jsonify({'description':'Check user'})

# Приложение 6

import requests

import sys

from keylogger import KeyLogger

def is\_registred(server\_url, user):

    answer = requests.post(server\_url+':5000/is\_registered', json={'user':user}).json()

    return True if answer['status']=='is\_exist' else False

def registration(server\_url, user):

    keylogger = KeyLogger(user)

    registration\_data = keylogger.registration()

    answer = requests.post(server\_url+':5000/new\_client', json=registration\_data).json()

    if answer['status'] == 'successfully':

        print('\n[+] Удачная регистрация')

    else:

        print('\n[+] Неудачная регистрация')

def check\_user(server\_url, user):

    keylogger = KeyLogger(user)

    capture\_data = keylogger.keys\_capture()

    answer = requests.post(server\_url+':5000/check\_client', json=capture\_data).json()

    print(answer['result'])

def main():

    server\_url = sys.argv[1]

    user = input('Введите ваше имя: ')

    if not is\_registred(server\_url, user):

        registration(server\_url, user)

    else:

        print('[+] Пользователь уже существует в системе.')

    while True:

        check\_user(server\_url, user)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

# Приложение 7

from pynput import keyboard

import time

import random

special\_keys = (keyboard.Key.esc, keyboard.Key.shift\_l, keyboard.Key.alt\_l, keyboard.Key.ctrl\_l, keyboard.Key.enter, keyboard.Key.delete,

                keyboard.Key.backspace, keyboard.Key.caps\_lock, keyboard.Key.cmd, keyboard.Key.down, keyboard.Key.end, keyboard.Key.left,

                keyboard.Key.num\_lock, keyboard.Key.page\_down, keyboard.Key.page\_up, keyboard.Key.up, keyboard.Key.print\_screen,

                keyboard.Key.f1,keyboard.Key.f2,keyboard.Key.f3,keyboard.Key.f4,keyboard.Key.f5,keyboard.Key.f6,keyboard.Key.f7,

                keyboard.Key.f8,keyboard.Key.f9,keyboard.Key.f10,keyboard.Key.f11,keyboard.Key.f12,keyboard.Key.f13,keyboard.Key.f14,

                )

train\_text\_list = list(range(1, 7))

class KeyLogger():

    def \_\_init\_\_(self, name) -> None:

        self.name = name

        self.keys = {

            'user': name,

            'data': []

        }

    def get\_key\_id(self, key):

        try:

            return key.vk

        except AttributeError:

            return key.value.vk

    def on\_press(self, key):

        try:

            if not (self.keys['data'][-1]['event'] == 'Down' and self.get\_key\_id(key) == self.keys['data'][-1]['keycode']):

                if key not in special\_keys:

                    self.keys['data'].append({'keycode': self.get\_key\_id(key), 'event': 'Down', 'time': time.time()})

        except:

            if key not in special\_keys:

                self.keys['data'].append({'keycode': self.get\_key\_id(key), 'event': 'Down', 'time': time.time()})

    def on\_release(self, key):

        if key not in special\_keys:

            self.keys['data'].append({'keycode': self.get\_key\_id(key), 'event': 'Up', 'time': time.time()})

    def print\_registration\_information(self):

        selected\_samples = random.sample(train\_text\_list, 2)

        for n in selected\_samples:

            print('Введите следующий текст. В конце нажмите клавишу Enter.')

            f = open(f"text\_for\_train/{n}.txt", "r")

            print(f.read())

            input()

            f.close()

    def registration(self):

        listener = keyboard.Listener(on\_press=self.on\_press, on\_release=self.on\_release)

        listener.start()

        try:

            listener.wait()

            self.print\_registration\_information()

        finally:

            listener.stop()

        return self.keys

    def is\_ready\_to\_send(self):

        while True:

            if len(self.keys['data'])>34 and self.keys['data'][-1]['event']=='Up':

                return

    def keys\_capture(self):

        listener = keyboard.Listener(on\_press=self.on\_press, on\_release=self.on\_release)

        listener.start()

        try:

            listener.wait()

            self.is\_ready\_to\_send()

        finally:

            listener.stop()

        return self.keys

if \_\_name\_\_=='\_\_main\_\_':

    test = KeyLogger('test')

    info = test.keys\_capture()

    print(info['data'])