Санкт-Петербургский государственный университет

Платонова Екатерина Алексеевна

Выпускная квалификационная работа

Получение данных из облачного хранилища VK

Уровень образования: бакалавриат

Направление *02.03.03 «Математическое обеспечение и администрирование информационных систем»*

Основная образовательная программа *СВ.5006.2018 «Математическое обеспечение и администрирование информационных систем»*

Научный руководитель:
доц. каф. СП, к.т.н. Ю.В. Литвинов

Консультант:
программист ООО “Белкасофт” А. И. Цой

Рецензент:
инженер-программист ООО “Белкасофт” А.А.Еремин

Санкт-Петербург
2022

Saint Petersburg State University

Platonova Ekaterina Alekseevna

Bachelor's Thesis

Getting data from VK cloud storage

Education level: bachelor

Speciality *02.03.03 "Software and Administration of Information Systems"*

Programme *CB.5006.2018 "Software and Administration of Information Systems"*

Scientific supervisor:
Software Engineering Chair, C.Sc Y.V. Litvinov

Consultant:
Software Engineer at "Belkasoft" A. I. Tsoy

Reviewer:
Software Engineer at "Belkasoft" A.A. Eremin

Saint Petersburg
2022

Оглавление

Введение

Сегодня невозможно представить жизнь человека без различных цифровых устройств. Используя возможности гаджетов, люди ежедневно обмениваются сообщениями, фотографиями, файлами, аудио- и видеодокументами. Информация такого типа может быть полезна для цифровой криминалистики. Объем передаваемых данных является большим, поэтому анализировать данные вручную оказывается затруднительным. Шифрование информации, используемое для обеспечения безопасности пользователей в социальных сетях и мессенджерах, также создает препятствие для анализа. При этом данные могут храниться разными способами: в облачных хранилищах, во внутренней памяти устройств, на внешних накопителях. Специалистам в области цифровой криминалистики необходимы инструменты для простого и быстрого извлечения информации из хранилищ. В данной работе будут реализованы три способа получения данных из облачного хранилища VK.

VK [1] является лидером среди социальных сетей в России [27]. Проектов, в которых реализуется извлечение различных данных из ВКонтакте, немного, и они либо являются коммерческими с закрытым исходным кодом, либо содержат устаревшую информацию. Различные статьи, находящиеся в свободном доступе, в большинстве случаев не содержат в себе полезных для исследования материалов. Вследствие этого, необходимо самим реализовать методы для извлечения данных. Появляются следующие задачи: изучить клиент-серверное взаимодействие, работу социальной сети ВКонтакте, трафик приложения ВКонтакте, его взаимодействие с различными устройствами.

Компания «Белкасофт» производит разработку инструмента Belkasoft X [15], который позволяет получать разнообразную информацию из различных источников, быстро анализировать и делиться с другими экспертами. Компанией была поставлена задача исследовать и реализовать методы извлечения данных из облачного хранилища системы ВКонтакте. Для того, чтобы в будущем интегрировать

[1] Далее по тексту – ВКонтакте

данный инструмент в проект Belkasoft X, реализация методов будет происходить на языке программирования C#.

Получение данных из облачного хранилища требует прохождения авторизации. Для этого можно использовать один из следующих способов:

- с помощью логина и пароля,
- с помощью токена авторизации,
- с помощью файлов cookies.

Так как логин и пароль могут быть не всегда известны, то нужны альтернативные способы преодоления этапа авторизации. Так, например, с помощью ключа доступа, хранящегося на устройстве, можно успешно реализовать работу методов получения данных из социальной сети. Другим способом может послужить авторизация с использованием cookie, которые создаются при входе в аккаунт с использованием браузера.

В данной работе будет рассмотрено три метода извлечения информации:

1. через API ВКонтакте стандартными средствами;
2. путем имитации работы браузерного клиента для системы ВКонтакте;
3. через имитацию работы системы ВКонтакте на мобильном устройстве.

1. Постановка задачи

Целью данной работы является реализовать способы для извлечения данных из облачного хранилища “ВКонтакте” – списков чатов, сообщений, файлов, документов, фотографий и другой информации. Для этого необходимо решить следующие задачи.

1. Выполнить обзор существующих инструментов извлечения данных из ВКонтакте.
2. Реализовать извлечение данных с помощью публичного API ВКонтакте.
3. Произвести вход в аккаунт и извлечь данные путем имитации работы веб-клиента.
4. Получить доступ к зашифрованному трафику устройства и проанализировать его.
5. Реализовать авторизацию и извлечение данных путем имитации работы мобильного клиента ВКонтакте.
6. Провести апробацию.
7. Интегрировать результаты в продукт Belkasoft X.

2. Обзор

2.1. Инструменты и технологии

2.1.1. Библиотека Vk.Net

Vk.Net [12] — клиентская библиотека, написанная на языке программирования C#, для работы с API ВКонтакте. Она была выбрана как самая удобная для C#. В ней содержится множество методов для работы с данными. В ходе работы были использованы методы, перечисленные ниже.

- *Authorize* — позволяет получить по логину и паролю AccessToken, который необходим для работы методов извлечения данных.
- *Users.Get* — позволяет получить информацию о пользователе.
- *Friends.Get* — возвращает список идентификаторов друзей пользователя и расширенную информацию о его друзьях.
- *Groups.Get* — позволяет получить список групп пользователя.
- *Video.Get* — позволяет получить список видеозаписей.
- *Docs.Get* — позволяет получить ссылки на документы пользователя.
- *Photo.GetAll* — позволяет получить ссылки на фотографии пользователя.

Существуют и другие библиотеки для работы с методами API. Например, InTouch — библиотека на C# [4]. Библиотеки vk [18] и vk_api [19] обеспечивающие работу с API ВКонтакте на языке Python.

Библиотека VK Java SDK (Software Development Kit) [16] является официальным клиентом ВКонтакте для работы с API ВКонтакте на языке программирования Java, а VK API Lib [17] — библиотка, предназначенная для работы с API ВКонтакте на C++. На основе данной

библиотеки можно создавать свои классы для работы с различными секциями данных ВКонтакте, а не использовать только существующие методы.

2.1.2. Протокол OAuth 2.0

OAuth 2.0 [7] — протокол авторизации, позволяющий выдать одному сервису права на доступ к ресурсам другого. Например, вместо регистрации на сайте можно авторизоваться с помощью уже имеющегося личного аккаунта в социальной сети и продолжить работу от имени владельца данного аккаунта. Это избавляет пользователей от ввода логина и пароля.

Результатом авторизации является access token — ключ, представляющий собой набор символов и являющийся пропуском к личной информации. При обращении к данным токен передается в качестве параметра запроса, либо указывается в его заголовках [8].

2.1.3. DevTools

Chrome DevTools — набор инструментов, позволяющих создавать, тестировать и отлаживать веб-сайты. Современные браузеры, в данном случае Google Chrome, имеют встроенные инструменты разработчика, которые позволяют проверять сетевой трафик, потребляемый сайтом, его быстродействие.

2.1.4. Fiddler

Fiddler [2] — инструмент, выступающий не только в качестве анализатора трафика, но и прокси-сервера. Он позволяет отслеживать HTTP/HTTPS трафик, входящий и исходящий из браузеров и настольных приложений. Также, Fiddler позволяет создавать, модифицировать и имитировать HTTP/HTTPS запросы.

Еще одним известным инструментом для захвата и анализа сетевого трафика является Wireshark [14]. В отличие от Fiddler, позволяет искать ошибки и выявлять причины этих ошибок на уровнях стека

TCP/IP.

2.1.5. Jailbreak

Джейлбрейк — процесс по получению повышенных прав доступа на устройствах с операционной системой iOS [5]. После проведения данной операции появляется доступ к приватным сегментам файловой системы и возможность устанавливать специальные приложения, позволяющие менять и настраивать системные функции. В основном такие приложения скачиваются из каталога Cydia, который является альтернативой Apple App Store.

2.1.6. Протоколы SSL/TLS

SSL (Secure Sockets Layer) — протокол, осуществляющй безопасное соединение между сервером и клиентом. SSL предоставляет:

- шифрование — защита данных при передаче;
- аутентификация — процесс проверки данных на подлинность сервером;
- целостность — неприкосновенность запрашиваемых и отправляемых данных.

SSL используется в основе сетевого протокола HTTPS для зашифрованной передачи данных и аутентификации, и активируется при установке SSL-сертификата. Так, при загрузке сайтов сервер отправляет информацию об SSL-сертификате и, если проверка подлинности пройдена, то устанавливается безопасное соединение [9].

TLS (Transport layer security) является более усовершенствованной версией SSL.

SSL Pinning

SSL Pinning [10] — защита, при которой SSL-сертификат внедряется в код мобильного приложения. Таким образом при установке HTTPS-соединения, проверка осуществляется с сертификатом, закрепленном в

приложении, а не на устройстве. Такой способ защиты помогает предотвратить популярные атаки MITM [2].

При подмене сертификата создается два соединения: с сервером и с клиентом, при котором обе стороны не подозревают о вторжении, считая, что общаются друг с другом. Таким образом весь трафик проходит через устройство "человека по середине".

Хранилище сертификатов — файл базы данных, где диспетчер цифровых сертификатов (DCM) держит цифровые сертификаты. В DCM может находиться несколько хранилищ, каждое из которых под паролем.

Процесс проверки сертификата состоит в следующем.

- При подключении клента к веб-серверу, находящемуся под защитой SSL, клиент запращивает информацию об SSL-сертификате и публичный ключ.
- Клиент проверяет данные, зашифровывает сгенерированный сеансовый ключ и отправляет на сервер.
- Сервер расшифровывает сеансовый ключ. Устанавливается безопасное соединение.

2.2. Существующие решения

- Оксиджен Софтвер «Мобильный Криминалист» — инструмент для извлечения и анализа данных из различных устройств и сервисов. Получить доступ к данным можно двумя способами [21]: по логину и паролю или по токену.

  «Мобильный Криминалист» извлекает из ВКонтакте следующие данные:

  – информацию об аккаунте;

  – контакты;

[2]MITM (Man in the middle) — разновидность атаки, заключающейся в перехвате трафика с целью сбора, искажения или удаления данных.

  - сообщения (личные и групповые чаты);
  - сообщества (группы и встречи);
  - вложения (фото, видео, аудио, документы и др.);
  - посты на стене владельца;
  - фотографии;
  - видео (информация об объекте и ссылка, по которой можно открыть видео);
  - аудио (информация об объекте и ссылка, по которой можно открыть аудио);
  - документы владельца.

  Информация о реализации методов извлечения данных в открытом доступе отсутствует.

- Различные расширения для браузера.
  - VK Music download — программа для скачивания аудиозаписей ВКонтакте. Существует открытый код на Python [25], JavaScript [26], C# [24].
  - VKSaver [11] — программа для скачивания аудиозаписей и видеозаписей ВКонтакте.
  - VK Video saver — программа для скачивания видеозаписей ВКонтакте. Существует открытый код на JavaScript [23].
  - Voiceload — бот [3] для скачивания голосовых сообщений ВКонтакте. Отправляет ссылку на скачивание сообщения, только после того, как сообщение переслано в бота.

- В статьях [?] [22], описан способ извлечения данных с помощью API ВКонтакте. В статье [13] получение данных осуществляется с целью сбора информации о деструктивном контенте, публикуемом

[3]Интернет-бот — программа, автоматически выполняющая некоторый набор задач, запрос на которые приходит от пользователей.

на страницах пользователей и в сообществах. В работе приводится пример составления запроса вручную и ответ на него.

В статье [22] получают информацию для разработки рекомендательной системы. Подробно реализация сбора информации не описывается, приводится пример ответа в формате JSON на запрос получения аудиозаписей.

Открытые решения — хороший пример, но они не гарантируют, что выполняется только скачивание данных — информация может отправляться на другие сервера, что критично для цифровой криминалистики. Также, задача получения данных из облачных хранилищ имеет важный нюанс — периодические изменения: усложнение процессов скачивания данных, изменение алгоритмов двухфакторной авторизации. Это говорит о том, что данная работа не потеряет свою ценность благодаря проведенным в ней исследованиям.

3. Архитектура

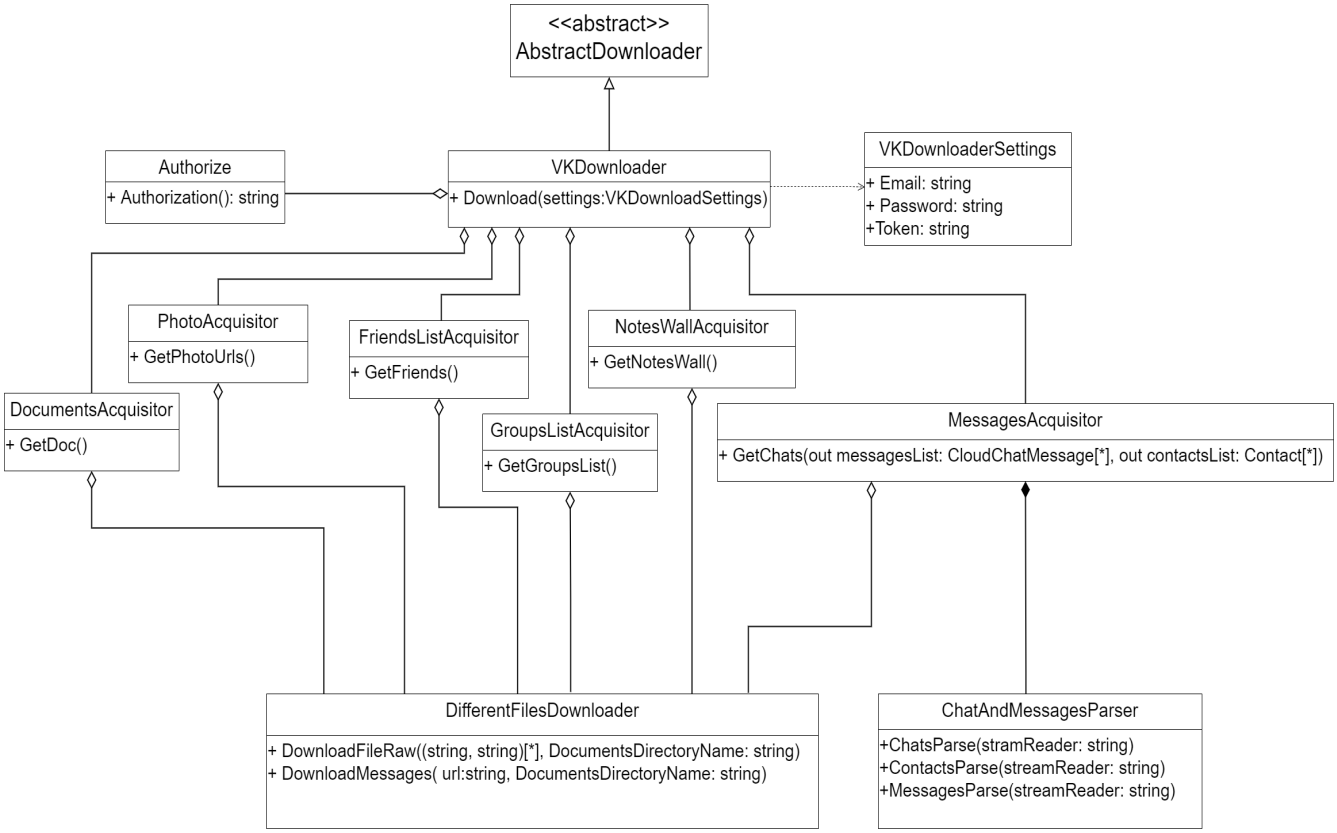

Рис. 1: Диаграмма классов

Был реализован прототип инструмента по извлечению данных из социальной сети ВКонтакте на языке C#. В него входят три способа извлечения данных, работа которых представлена на диаграмме 1.

Класс VKDownloader наследуется от абстрактного класса AbstractDownloader, который является частью архитектуры продукта Belkasoft X.

Класс VKDownloaderSettings содержит логин, пароль и токен авторизации. Объект этого класса передается в качестве параметра в один из методов класса VKDownloader.

Authorize — класс, в котором реализован процесс авторизации и двухфакторной авторизации. В Authorize содержатся методы, осуществляющие отправку различных запросов, имитирующих этап входа в аккаунт.

Классы MessagesAcquisitor, DocumentsAcquisitor,

FriendsListAcquisitor, PhotoAcquisitor, NotesWallAcquisitor, GroupsListAcquisitor отвечают за извлечения разного типа данных из аккаунта пользователя. В каждом классе содержатся методы, реализующие запросы на получение данных. В некоторых классах осуществляется и сбор необходимой информации из ответов.

CloudChatMessage — класс, содержащий в себе различную информацию о сообщениях и чатах, а именно: id чата, текст сообщения, id отправителя, ссылки для скачивания различных данных, тип сообщения, список уникальных id для вложений из сообщений, информацию о геолокации.

Contact — класс, хранящий информацию о друзьях пользователя и активных участниках бесед: имена и фамилии, домашний и рабочий адреса, id аккаунта, мобильный, домашний или рабочий телефоны, аккаунты в различных социальных сетях и другое.

ParserChatAndMessages — класс, в котором происходит сбор id чатов и сообщений, активных участников диалогов, тексты сообщений, информация о вложениях сообщений.

DifferentFilesDownloader — класс, отвечающий за скачивание различных файлов из сообщений и аккаунта.

4. Особенности реализации

4.1. Получение данных с помощью программного интерфейса API ВКонтакте

API ВКонтакте — инструмент, позволяющий извлекать различную информацию из базы данных социальной сети путем составления HTTPS-запросов к серверу. Для работы с методами интерфейса нужен access token — ключ доступа, для получения которого используется открытый протокол OAuth 2.0. Ключ доступа получался самым простым и коротким путем — Implicit Flow [3]. При таком способе, происходит перенаправление на URL-адрес, в фрагменте которого содержится access token. Implicit Flow обычно используется в тех случаях, когда требуется временный доступ к данным пользователя (токен действует несколько часов).

Существует еще и другой метод получения ключа, двухэтапный — Authorization code flow. При таком способе токен приходит после отправки GET-запсроса с кодом авторизации в качестве параметра.

Перед началом работы с публичным API было необходимо провести процедуру создания приложения на сайте ВКонтакте для получения ID приложения, который в последующем передается как параметр запроса. Обращение к методам API ВКонтакте реализовывалось с использованием библиотеки Vk.Net.

Авторизация состоит из двух этапов:

1. Ввод логина и пароля, для получения токена.

2. Авторизация с использованием токена.

Ниже приведен пример простой авторизации с использованием метода Authorize из библиотеки Vk.Net

```
static void Main(string[] args)
{
    var api = new VkApi();

    api.Authorize(new ApiAuthParams
```

```
        {
            ApplicationId = 123456, // ID приложения
            Login = ''Login'',
            Password = ''Password'',
            Settings = Settings.All // Права доступа приложения
        });
        Console.WriteLine(api.Token);
    }
```

Со временем получение ключа доступа способом Implicit Flow перестало работать по причинам, связанным с изменением алгоритмов авторизации. Эта проблема была решена следующим образом: для получения бессрочного Access Token нужно перейти на сайт, представленный ниже и выполнить инструкции: выбрать приложение, ввести логин и пароль от аккаунта, скопировать часть адресной от *access_token=*

```
Url: ''https://vkhost.github.io/''
```

Данную ссылку можно найти в документации к библиотеке Vk.Net [12].

После реализации всех методов в коде, с помощью программного интерфейса API ВКонтакте была получена информация о пользователе и его друзьях, фотографии и видео, списки друзей, групп и документов.

Перед использованием некоторых методов (например, для получения сообщений) необходимо написать в поддержку социальной сети для получения тестового или полного доступа к методам, который выдается не всем [20].

Чтобы получить доступ к методам, в письме необходимо приложить следующие данные:

- ссылку на скачивание приложения, если есть;
- видео работы приложения;
- описание функциональности;
- id приложения;
- сроки разработки.

4.2. Реализация веб-клиента для извлечения данных из ВКонтакте

Как было описано ранее, через публичный API ВКонтакте некоторые из данных недоступны. Поэтому нужны другие методы получения приватных данных из аккаунта пользователя. Один из способов — реализовать веб-клиент, чтобы общение системы ВКонтакте происходило не со сторонним приложением, а с привычным ей клиентом.

Для исследования работы браузера был выбран DevTools, как самый удобный и простой набор инструментов для анализа веб-трафика. Его возможностей было достаточно, чтобы проанализировать запросы, которые отправляются на сервер во время авторизации, загрузки чатов, сообщений, вложений и других полезных с точки зрения криминалистики данных. Также было исследовано формирование различных запросов и ссылок (например, для скачивание файлов) и какие параметры для этого нужны.

Использовалось два вида запросов: POST и GET. Каждый из которых содержит в себе: URL-адрес, на который нужно посылать запрос, статус ответа, заголовки запросов и ответов, параметры. Все необходимые части запроса можно посмотреть в DevTools.

4.2.1. Авторизация с известным логином и паролем

Успешная авторизация позволяет получить файлы cookies, необходимые для корректной работы последующих методов. Для реализации этого этапа необходимо формировать различные запросы: с известными и неизвестными заранее параметрами. Для получения значений неизвестных параметров нужно было исследовать, что они означают, как формируются, где хранятся. Например, в запросе для авторизации такими параметрами являются: ip_h, ig_h, ig_domain, to.

Значения указанных параметров можно найти в ответе на GET-запрос по следующему адресу:

```
Url: ''https://vk.com/login''
```

Также, в качестве значений параметров передаются известные логин и пароль.

После того, как значения были найдены, формировался POST-запрос на адрес:

```
Url: ''https://login.vk.com/?act=login''
```

с указанием необходимых заголовков и параметров. Пример POST-запроса представлен ниже.

```
var request = new HttpRequestMessage(HttpMethod.Post, ''https://login.vk.com/?act=login'');
request.Headers.Add(''Accept'', ''text/html,application/xhtml+xml,application/xml;q=0.9'');
    //заголовки запроса
     request.Headers.Add(''cache-control'', ''max-age=0'');
     request.Headers.Add(''upgrade-insecure-requests'', ''1'');
     request.Headers.Add(''sec-fetch-site'', ''same-site'');
     request.Headers.Add(''origin'', ''https://vk.com'');
     request.Headers.Add(''Referer'', ''https://vk.com/'');
     request.Headers.Add(''sec-fetch-user'', ''?1'');
    //параметры запроса
    request.Content = new FormUrlEncodedContent(new Dictionary<string, string>
    {
        [''act''] = ''login'',
        [''role''] = ''al_frame'',
        [''to''] = to,
        [''_origin''] = ''https://vk.com'',
        [''ip_h''] = ip_h,
        [''lg_h''] = lg_h,
        [''lg_domain_h''] = lg_domain,
        [''email''] = _login,
        [''pass''] = _password
    });
```

4.2.2. Двухфакторная авторизация

Если на аккаунте пользователя ВКонтакте включена двухфакторная авторизация, то в ответе на POST-запрос по адресу:

```
Url: ''https://login.vk.com/?act=login''
```

приходят необходимые cookies для формирования GET-запроса на адрес:

```
Url: ''https://vk.com/login?act=authcheck''
```

При его отправке на сервер, пользователю приходит код авторизации, а в ответе содержится парметр, представляющий собой строку из цифр и латинских букв. При формировании следующего запроса полученный параметр используется как часть одного из заголовков.

Завершающим этапом двухфакторной авторизации является отправка POST-запроса с известными кодом и полученным ранее параметром по адресу:

```
Url: ''https://vk.com/al_login.php?act=a_authcheck_code''
```

4.2.3. Ввод CAPTCHA

Капча (CAPTCHA — Completely Automated Public Turing test to tell Computers and Humans Apart — автоматически генерируемый тест, используемый для определения пользователя системы.

Сначала нужно проверить, содержится ли в ответе на завершающий двухфакторную авторизацию запрос ссылка на картинку с кодом. Если URL-адрес присутствует, то происходит скачивание и ввод пользователем кода. Далее отправляется запрос, где в качестве параметров выступают код с картинки, полученный при двухфакторнй авторизации параметр и специальный номер картинки, содержащийся в ссылке.

4.2.4. Получение пользовательских данных

При извлечении личных данных из аккаунта пользователя формируются POST/GET-запросы на различные URL-адреса. Так, например, для получения списка диалогов нужно отправить POST-запрос по адресу:

```
Url: ''https://vk.com/al_im.php?act=a_get_dialogs''
```

В ответе содержится JSON, из которого нужно взять только имя отправителя и номер диалога. Далее из каждого чата извкаются сообщения. Для этого отправляется POST-запрос по адресу:

```
Url: ''https://vk.com/al_im.php?act=a_history''
```

Некоторые сообщения не содержат текста, а представляют собой другие виды данных: стикер [4], голосовое сообщение, аудиозапись, файл и другое. В этом случае, необходимо вывести подробную информацию о сообщении:

- время отправления и получения,
- имя отправителя,
- тип сообщения,
- ссылки на файлы,
- транскрипт и ссылку на скачивание для голосовых сообщений.

В ответ на запрос приходит JSON с различной информацией и HTML-кодом с ссылками на скачивание голосовых сообщений.

Следующий шаг — отправление POST-запроса на получение ссылок для скачивания фотографий по адресу:

```
Url: ''https://vk.com/al_photos.php?act=show_albums''
```

В ответе так же содержится JSON с HTML-кодом, откуда извлекаются ссылки.

Чтобы получить список групп и список друзей, формируются POST-запросы на соответсвующие URL-адреса:

```
Url: ''https://vk.com/al_groups.php?act=get_list''

Url: ''https://vk.com/al_friends.php?act=load_friends_silent''
```

В ответах содержится JSON, в котором интересными данными для групп являются названия сообществ, фотографии и ссылки. Для друзей — id страниц спользоватля, главное фото аккаунта, имя и фамилию.

При работе с файлами в социальной сети "ВКонтакте" возможны две ситуации, перечисленные ниже.

1. При нажатии на документ сразу происходит его скачивание.

[4] Стикер — картинка, выражающая какую-либо эмоцию

2. При нажатии документ открывается в новой вкладке, где его можно редактировать и скачивать.

Для получения документов при первом случае нужно сформировать GET-запрос на адрес:

```
Url: ''https://vk.com/docs''
```

В ответ приходит HTML-код, где в одной из строк находится JSON, содержащий id файла и ссылки для скачивания.

Если же встречается второй сценарий, то нужно исследовать, как браузер формирует ссылку на файл при нажатии на кнопку скачивания. Получение документов организовано в два этапа.

1. Отправление GET-запроса по адресу:

```
Url: ''https://vk.com/docXXX_YYY?wnd=1&fragment=0''
(XXX - id страницы пользователя, YYY - номер файла)
```

2. В ответе содержится следующий URL-адрес:

```
Url: ''https://vk.com/docXXX_YYY''
(XXX - id страницы пользователя, YYY - номер файла)
```

   на который отправляется еще один GET-запрос, после чего в ответе приходит ссылка на скачивание документа.

Завершающим этапом в извлечении пользовательских данных является получение записей со стены. Для этого формируется POST-запрос на адрес:

```
Url: ''https://vk.com/al_wall.php?act=get_wall''
```

Данные в ответе представлены в формате JSON с HTML-кодом, в котором содержится различная информация о записях:

- является ли запись закрепленной;
- является ли запись репостом;
- имя автора;

- текст;
- время записи;
- ссылка на обложку видео;
- ссылка на фотографии.

Если запись представляет собой видеоконтент, то в ответе присутствует и ссылка на скачивание видео, но находится она отдельно от остальной информации.

4.2.5. Реализация работы методов с использованием cookies

Способ извлечения данных через API стандартного браузерного клиента позволяет реализовать корректную работу методов не только через логин и пароль, но и с помощью cookies, которые можно получить из браузера пользователя.

В ходе исследования было выявлено, что необходимыми для получения ответов с интересующей информацией являются *remixsid* и *remixnsid*, которые нужно добавлять при формировании запросов следующим образом:

```
var adress = new Uri(''https://vk.com'');
_cookieContainer.Add(adress, new Cookie(''remixsid'', cookie_1));
_cookieContainer.Add(adress, new Cookie(''remixnsid'', cookie_2));
```

4.3. Имитация работы мобильного клиента ВКонтакте для извлечения данных

Для анализа трафика мобильного приложения ВКонтакте использовались Fiddler и устройство на iOS с jailbreak. Аналогично прошлому способу, отслеживались исходящие от клиента запросы и реализовывались в коде. Отличие состоит в необходимости обхода защиты SSL pinning, который не позволяет считывать и исследовать исходящие данные. Реализация процесса jailbreak и чтение трафика с помощью Burp Suite [1] описаны в статье [6].

4.3.1. Перехват HTTPS-трафика мобильного устройства

Чтобы начать перехватывать трафик, нужно установить сертификат Fiddler на устройство, затем настроить прокси в настройках Wi-Fi.

Обход SSL pinning

Теперь Fiddller отображает HTTP запросы, но этого недостаточно для реализации извлечения данных. Необходимо, чтобы перехватывались и HTTPS запросы, но есть препятствие — SSL pinning.

Реализация обхода осуществлялась с помощью SSL Kill Switch 2 [5]. В каталоге Cydia нужно найти и установить MTerminal и пакеты для обхода SSL-pinning. Далее в MTerminal нужно осуществить следующую последовательность команд:

1. команда su (Substitute/Super User) — для переключения на корневого пользователя [6];
2. пароль — alpine;
3. команда ls (List files in the directory) — для просмотра файлов;
4. команда dpkg(Debian package) -i (install) <имя_пакета>.deb — для загрузки пакета.

После такой последовательности действий пакет SSL.deb установливается на устройство. После его активации в настройках Fiddler может перехватывать любой трафик iOS-приложения.

4.3.2. Авторизация

Процесс авторизации заключается в основном в отправке GET-запросов. На финише в ответе приходит access token, который в дальнейшем небходимо указывать в заголовке «Authorize».

Первый GET-запрос отправляется по адресу:

[5]SSL Kill Switch 2 — инструмент для отключения проверки SSL-сертификата, включая закрепление сертификата, в приложениях iOS и macOS.

[6]Корневой пользователь — пользователь, имеющий общесистемные права доступа высокого уровня. Например, пользователь root может вносить изменения в файлы других пользователей или просматривать большую часть из них.

```
URL: ’’https://clientapi.mail.ru/fcgi-bin/libverifysettings?action_type=default’’
```

где через & нужно указать следующие параметры:

- application=VK,
- application_id,
- capabilities=call_number_fragment%2Cping_v2,
- device_id,
- language=en,
- libverify_build=165
- libverify_version=2.0.0,
- os_version=13.3.1,
- platform=ios,
- system_id,
- use_apns_sandbox=0,
- version=2774,
- signature.

Следующий GET-запрос отправляется по адресу:

```
URL: ’’https://clientapi.mail.ru/fcgi-bin/verify?application=VK’’
```

где через & нужно указать не только перечисленные ранее параметры, но и те, что ниже.

- auth_type=VKCONNECT,
- checks=vkc,
- device_name,

- service=vk_registration
- session_id.

Завершающим этапом авторизации является формирование GET-запроса на адрес:

```
URL: ’’https://api.vk.com/oauth/token?2fa_supported=1’’
```

где через & указываются следующие параметры:

- client_id,
- client_secret,
- device_id,
- external_device_id,
- grant_type=password,
- idfa,
- idfv,
- password,
- sak_version=1.71,
- scope=all,
- username,
- validate_session.

Перечисленные параметры используются для формирования GET-запросов и в двухфакторной авторизации, поэтому далее информация о параметрах опускается.

4.3.3. Двухфакторная авторизация

Двухфакторная авторизация состоит из этапов, перечисленных ниже.

1. GET-запрос по адресу:

   `URL: ''https://clientapi.mail.ru/fcgi-bin/verify?''`

   в ответе на который содержится информация о статусе операции и токен.

2. GET-запрос по адресу:

   `URL: ''https://api.vk.com/oauth/token?2fa_supported=1''`

   Код состояния такого запроса 401 — Unauthorized. При реализации необходимо обработать исключение, в тексте которого содержится необходимая информация, а именно: код проверки и адрес, на который необходимо отправить запрос.

3. GET-запрос по адресу:

   `URL: ''https://api.vk.com/method/auth.validatePhone?''`

   с указанием полученного ранее кода проверки. В ответе содержится информация, что код для прохождения двухфакторной авторизации отправлен.

4. GET-запрос по полученному из прошлого запроса URL-адресу.

5. GET-запрос по адресу:

   `URL: ''https://api.vk.com/oauth/token?2fa_supported=1''`

   В заголовках указываются: код, приходящий на номер телефона или в личные сообщения в социальной сети "ВКонтакте" или последние четыре цифры телефона, с которого совершается входящий звонок.

6. Если авторизация пройдена успешно, то в ответе содержится access token, необходимый для последующих запросов.

4.3.4. Извлечение пользовательских данных

Получение сообщений, фотографий, документов, списка групп, записей на стене происходит аналогично описанным действиям в главе 4.2.4.

5. Апробация

Для проведения апробации были взяты два аккаунта: тестовый, без двухфакторной авторизации, и личный, с двухфакторной.

На тестовый аккаунт были заранее загружены данные каждого типа и заведены несколько диалогов с различными сообщениями, где во вложениях содержатся: аудиозаписи, голосовые сообщения, истории, документы, видео, граффити и другое.

В личном аккаунте узнать точное количество сообщений и диалогов не представляется возможным, поэтому результаты по этим типам данных не проверить. Количество остальной информации отображается на странице пользователя.

После проведения апробации были составлены таблицы результатов 2 и 3, взглянув на которые можно сделать вывод, что количество загруженных данных в тестовом аккаунте и имеющихся в личном — совпали с количеством выгруженных данных (количество которых было известно).

	Диалоги		Сообщения		Файлы	
	извлечено	в аккаунте	извлечено	в аккаунте	извлечено	в аккаунте
Тестовый аккаунт (без двухфакторной авторизации)	4	4	54	54	15	15
Личный аккаунт (с двухфакторной авторизацией)	87	87	844 657	неизвестно	411	411

Рис. 2: Результат апробации 1

	Фотографии		Записи со стены		Друзья		Группы	
	извлечено	в аккаунте	извлечено	в аккаунте	извлечено	в аккаунте	извлечено	в аккаунте
Тестовый аккаунт (без двухфакторной авторизации)	7	7	5	5	2	2	20	20
Личный аккаунт (с двухфакторной авторизацией)	60	60	18	18	238	238	164	164

Рис. 3: Результат апробации 2

5.1. Сравнение методов

Согласно результатам, представленным в таблицах 4 и 5, были сделаны выводы, что программный интерфейс API ВКонтакте имеет значительне минусы, а именно:

- требуется регистрация приложения;
- есть ограничения на число запросов;
- есть ограничения на доступ к некоторым методам;

Среди плюсов — наличие актуальной документации, которой нет для других способов извлечения данных.

Для API стандартного браузерного клиента плюсами являются: возможность использования хранящихся в браузере cookies, с помощью которых можно обойти этап авторизации и извлечь данные любого типа. Среди минусов — ограничение на число запросов.

Извлечение данных с помощью имитации работы мобильного клиента ВКонтакте позволяет так же получить любые личные данные пользователя. Кроме того, с помощью хранящегося на устройстве ключа, можно авторизоваться в аккаунте не используя логин и пароль.

	API ВКонтакте	API стандартного браузерного клиента	API клиента ВКонтакте
Диалоги	-	+	+
Сообщения	-	+	+
Файлы	+	+	+
Фотографии	+	+	+
Записи со стены	+	+	+
Список друзей	+	+	+
Список групп	+	+	+

Рис. 4: Сравнительная таблица 1

5.2. Результат интеграции в Belkasoft X

Продукт Belkasoft X предназначен для компьютерной, мобильной и облачной криминалистики. С его помощью можно быстро собрать

	Публичное API ВКонтакте	API стандартного браузерного клиента	API мобильного клиента ВКонтакте
Извлечены все данные	нет	да	да
Документация	есть	нет	нет
Регистрация приложения	есть	нет	нет
Ограничение на число запросов	есть	нет	нет

Рис. 5: Сравнительная таблица 2

информацию с различных облачных хранилищ и, проанализировав их, составить отчет по найденным уликам.

Используя продукт Belkasoft X и интегрированный в него инструмент по получению данных из облачного хранилища ВКонтакте была выполнена процедура извлечения различной информации из тестового аккаунта в социальной сети.

Перед началом работы были использованы:

- логин и пароль для прохождения авторизации;
- приватный API мобильного приложения ВКонтакте.

Примеры работы инструмента, интегрированного в Belkasoft X, приведены на рисунках 6 и 7.

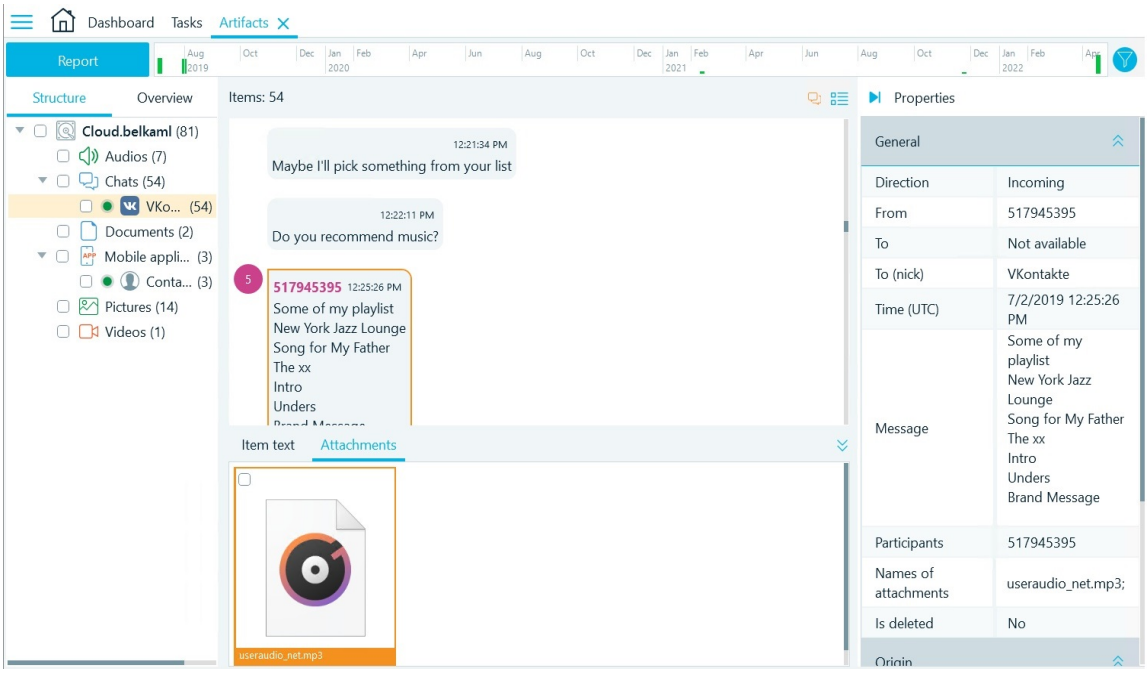

Рис. 6: Результат извлечения чатов из тестового аккаунта в Belkasoft X

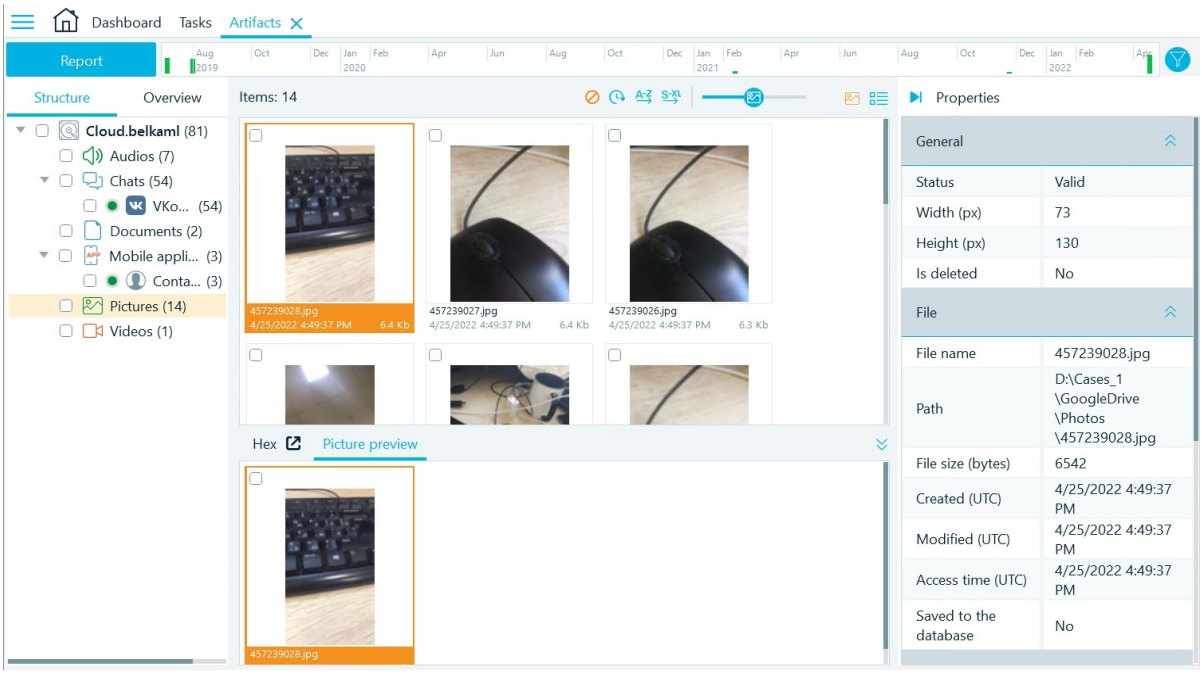

Рис. 7: Результат извлечения фотографий из тестового аккаунта в Belkasoft X

6. Заключение

В рамках данной работы были получены следующие результаты:

- Выполнен обзор существующих методов для извлечения данных

из социальной сети “ВКонтакте”.

- Реализована процедура извлечения данных из облачного хранилища “ВКонтакте” с помощью открытого программного интерфейса (API):
  - по токену,
  - логину и паролю.
- Реализована имитация веб-клиента для получения информации из “ВКонтакте”:
  - поддержаны авторизация и двухфакторная авторизация с известными логином и паролем;
  - поддержана авторизация с известными cookies.
- Реализована имитация мобильного клиента “ВКонтакте” для извлечения данных:
  - поддержаны авторизация и двухфакторная авторизация.
- Проведена апробация:
  - выполнена загрузка данных в тестовый аккаунт,
  - выполнена выгрузка данных из тестового и личного аккаунтов,
  - произведено сравнение результатов.
- Произведена интеграция результатов в продукт Belkasoft X.

Код проекта закрыт и принадлежит компании ООО ”Белкасофт”.

Список литературы

[1] Burp Suite // URL: https://portswigger.net/burp (дата обращения 14.03.2022).

[2] Fiddler // URL:https://www.telerik.com/fiddler/fiddler-classic (дата обращения: 29.11.2021).

[3] Implicit flow // URL: https://dev.vk.com/api/access-token/getting-started (дата обращения 08.02.2022).

[4] InTouch // URL: https://github.com/virtyaluk/InTouch (дата обращения 08.02.2022).

[5] Jailbreak // URL: https://elibrary.ru/download/elibrary_27489299_23073403.pdf (дата обращения: 15.01.2022).

[6] Jailbreak 12.4 and SSL pinning bypass // URL: https://medium.com/@yogendra_h1/ios-application-security-jailbreak-12-4-5e3fc0dc0726 (дата обращения 14.01.2022).

[7] OAuth 2.0 // URL: https://oauth.net/ (дата обращения: 29.11.2021).

[8] OAuth 2.0 простым и понятным языком // URL: https://habr.com/ru/company/vk/blog/115163/ (дата обращения: 29.11.2021).

[9] SSL // URL: https://ssl.com.ua/info/how-ssl-works/ (дата обращения 08.02.2022).

[10] SSL pinning // URL: https://habr.com/ru/company/surfstudio/blog/504914/ (дата обращения 14.01.2022).

[11] VKSaver // URL: https://github.com/RomanGL/VKSaver (дата обращения: 15.12.2021).

[12] Vk.Net. // URL: https://vknet.github.io/vk/ (дата обращения: 15.11.2021).

[13] Voiceload // URL:https://vk.com/voiceload (дата обращения: 29.11.2021).

[14] Wireshark // URL:https://www.wireshark.org/ (дата обращения: 29.11.2021).

[15] Белкасофт. // URL: https://belkasoft.com/ru/home/about.asp (дата обращения: 15.11.2021).

[16] Библиотека Java SDK для работы с API ВКонтакте на Java // URL:github.com/VKCOM/vk-java-sdk (дата обращения: 29.11.2021).

[17] Библиотека VK API Lib для работы с API ВКонтакте на C++ // URL:https://github.com/Kolsha/VK (дата обращения: 29.11.2021).

[18] Библиотека vk для работы с API ВКонтакте на Python // URL:https://vk.readthedocs.io/en/latest/ (дата обращения: 29.11.2021).

[19] Библиотека vk_api для работы с API ВКонтакте на Python // URL:https://github.com/python273/vk_api (дата обращения: 29.11.2021).

[20] Ограничение на получение сообщений через API Вконтакте. // URL: https://vk.com/dev/messages_api (дата обращения: 29.11.2021).

[21] Оксиджен Софтвер. // URL: https://www.oxygensoftware.ru/ru/features/clouds/270-vkontakte (дата обращения: 29.11.2021).

[22] Разработка рекомендательной системы на основе данных из профиля социальной сети «Вконтакте». // URL: https://elibrary.ru/item.asp?id=22092035 (дата обращения: 29.11.2021).

[23] Скачивание видео из ВКонтакте. Код на JavaScript. // URL: https://github.com/DaniilAfendulov/VK-video-dowload (дата обращения: 15.12.2021).

[24] Скачивание музыки из ВКонтакте. Код на C#. // URL: https://habr.com/ru/post/525346/ (дата обращения: 15.12.2021).

[25] Скачивание музыки из ВКонтакте. Код на Python. // URL: https://gist.github.com/st4lk/4708673 (дата обращения: 15.12.2021).

[26] Скачивание музыки из ВКонтакте без использования публичного API. Код на JavaScript. // URL: https://habr.com/ru/post/340810/ (дата обращения: 15.12.2021).

[27] Список популярных социальных сетей в России. // URL: https://ria.ru/20210706/sotsseti-1740025260.html (дата обращения: 15.11.2021).