Московский государственный университет имени М.В.Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра Системного Программирования

Лазарев Владимир Александрович

Исследование методов OSINT для поиска информации о человеке

Курсовая работа

Научный руководитель:
к.ф.-м.н.
Турдаков Денис Юрьевич
Научный консультант:
Яцков Александр Константинович

Москва, 2021

Аннотация

Исследование методов OSINT для поиска информации о человеке

Лазарев Владимир Александрович

Данная работа посвящена исследованию и разработке методов OSINT для поиска информации о человеке. Данная курсовая содержит описание реализованных методологий и повествует о созданных приемах извлечения информации.

В ходе работы были изучены и представлены существующие различные методы как по способу взаимодействия с сервисами: извлечение данных с web-страницы и посредством скрытого или открытого api; так и по типу сервиса: поисковый агрегатор и социальные сети.

Содержание

1 Введение

В современном мире присутствует огромное количество социальных сетей и поисковых ресурсов, которые имеют собственные стараницы в сети Интернет. Это могут быть различные социальные сети: от медиа (Instagram, TikTok), мессенджеров (Telegram, WhatsApp), так и полноценных, в которых можно указывать информацию о личности (ВКонтакте, Facebook, LinkedIn). И большинство людей имеют аккаунты сразу в нескольких социальных сетях одновременно, самостоятельно и по доброй воле делятся своими персональными данными.

Вместе с этим активно развиваются сервисы, которые делают подборку контента на основе агрегации и обработки данных, полученных из Интернет. Даже та самая контекстная реклама, которая старается продвинуть товары и услуги, которые недавно искались пользователем – есть часть тех самых сервисом и OSINT в целом [1]. Например YouTube отображает в рекомендованных видеозаписях тот контент, которых находится на стыке популярного сейчас и тех тематик, которые просматривали ранее. Также существуют сервисы по предоставлению персонализированных новостных лент, самой популярной в RU-сегменте является Яндекс.Дзен. У этого подхода есть существенные плюсы, такие как пользователь всегда будет актуальный и необходимый ему контент.

Говоря дальше об OSINT, стоит упомянуть, что точный термин ставится как «разведка на основе открытых источников». То есть, в сборе и обработки данных нет ничего противозаконного, так как никакие базы данных и устройства не взламываются. Но и этого количества информации весьма достаточно, чтоб иметь некую картину о пользователе сети Интернет или организации с активной социальной жизнью. С помощью данной технологии правительства всех стран могут отслеживать и поддерживать национальную безопасность, бороться с терроризмом и устанавливать слежку за участниками преступных группировок, оценивать настроения и взгляды общественности как внутри государства, так и вне ее.

Например, такое ПО как Palantir активно используется в полиции для отслеживания преступников, ведь оно агрегирует данные не только из Интернет, но и предоставляет картинку с камер видеонаблюдения и строит зависимости на географической карте страны.

Таким образом, появляется задача разработки обширной и автоматической системы

поиска и сбора данных из открытых источников сети Интернет, способных извлекать данные установленного формата из большого количества веб-ресурсов. Под обширностью понимается, что необходимо задействовать по максимуму все возможные поисковые ресурсы, ведь именно в них данные уже заранее проанализированы и структурированы. Под автоматизацией подразумевается то, что оператору системы необходимо будет только единожды настроить параметры сбора, а информация будет автоматически далее собираться, отсеивать дубликаты и сохраняться в базу данных.

В разделе 1 сформулирована постановка задачи. В разделе 2 приведен анализ существующих решений методов поиска, сбора и анализа информации из открытых источников. В разделе 3 описано исследование и построение решения задачи. В разделе 4 приведено описание практической части курсовой работы. В конце документа сформулировано заключение.

2 Постановка задачи

Целью данной курсовой работы является исследование и разработка методов OSINT для поиска информации о человеке. Для решения задачи, ее можно разбить на несколько подзадач: сбор информации при помощи поисковых сервисов, сбор информации с помощью социальных сетей. В свою очередь каждую из подзадач также можно поделить на следующие части: определение структуры web-страницы и извлечение данных непосредственно из страницы, поиск более быстрого доступа к информации посредством открытого или закрытого api.

В итоге для достижения постановленной цели необходимо решить следующие задачи:

- Поиск данных в поисковых сервисах:
  - Провести анализ литературы и существующих решений для извлечения данных из поисковых систем;
  - Разработать методы поиска и сбора информации из поисковых систем:
    - Проанализировать структуру web-страниц поискового сервиса;
    - Реализовать метод поиска и извлечения информации при помощи атрибутов web-страницы;
    - Провести исследование о возможности получения данных из ресурса посредством открытого или закрытого api;
    - Если api реализовано на стороне сервиса, то реализовать метод поиска и сбора посредством api;
  - Получить тестовые данные от реализованных методов и провести анализ, исследование полученной информации;
- Поиск данных в социальных сетях:
  - Провести анализ литературы и существующих решений для извлечения данных из социальных сетей;
  - Разработать методы поиска и сбора информации из социальных сетей:
    - Проанализировать структуру web-страниц социальных сетей;

    * Реализовать метод поиска и извлечения информации при помощи атрибутов web-страницы;
    * Провести исследование о возможности получения данных из ресурса посредством открытого или закрытого api;
    * Если api реализовано на стороне соц. сети, то реализовать метод поиска и сбора посредством api;
- Получить тестовые данные от реализованных методов и провести анализ, исследование полученной информации;

3 Обзор существующих решений

3.1 Поиск данных в поисковых сервисах

3.1.1 Google Dorks (Google Hacking)

Google Dorks[1] – это по сути та же самая поисковая система от Google. Отличие заключается только в том, что обычный пользователь вбивает типовые запросы а-ля «Какая погода в Москве?», то Google Dorks позволяет использовать специальные запросы для получения конкрентной информации. Google Dorks имеет множество операторов, которые можно использовать для составления очень гибких и точных запросов [2]. По факту, это запросы, с помощью которых можно проверить безопасность того или иного сайта, найти IP-адреса сервисов, камер. Весьма эффективна для поиска документации по ключевым словам, а также поиску людей с помощью тех же самых Google Dorks Queries.

Плюсы данный системы:

- быстрый и объемный поиск по ключевым словам.

Из недостатков системы можно определить следующее:

- составленный запрос выдаст перечень ссылок в интерфейсе поисковой системы, а не сами данные;
- перед использованием необходимо изучить синтаксис запросов;
- нет накопления собранной информации, нельзя отслеживать изменения (дельты);
- нет построения графа зависимостей объекта.

[1] https://www.google.com/

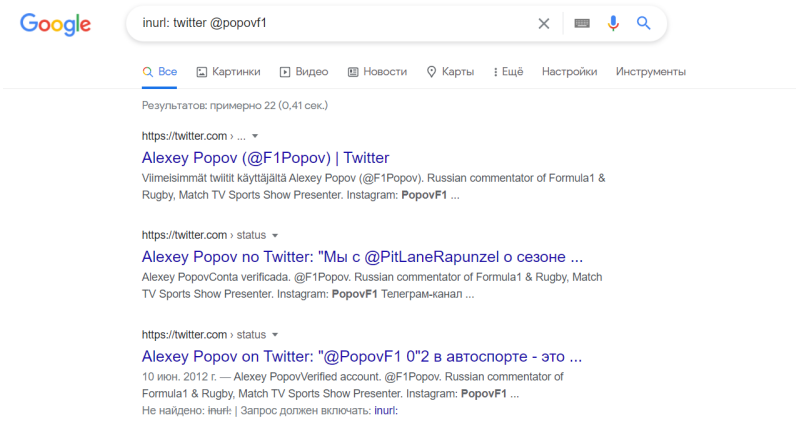

Рис. 1: Пример использования GDQ для поиска человека.

3.1.2 Carrot2

Carrot2 – движок кластеризации результатов поисковых запросов с открытым исходным кодом. Carrot2 может самостоятельно группировать по категориям найденные документы или данные. Работает в свою очередь как обычный поисковик, то есть по указанному ключевому слову возвращает некоторое множество ссылок, затем которые группируются по категориям [3].

Преимущества:

- быстрый и обширный поиск по ключевым словам;
- автоматическая группировка данных в соответствии с категориями;
- наличие удобного интерфейса с возможностью просмотра древовидной карты и круговидной диаграммы.

Недостатки:

- как и в случае с Google Dorks, Carrot2 возвращает нам перечень ссылок на источники данных, а не сами данные непосредственно;
- невозможно произвести точечный поиск файлов и данных, как это реализовано в Google Dorks. Как следствие – большое количество лишней информации.

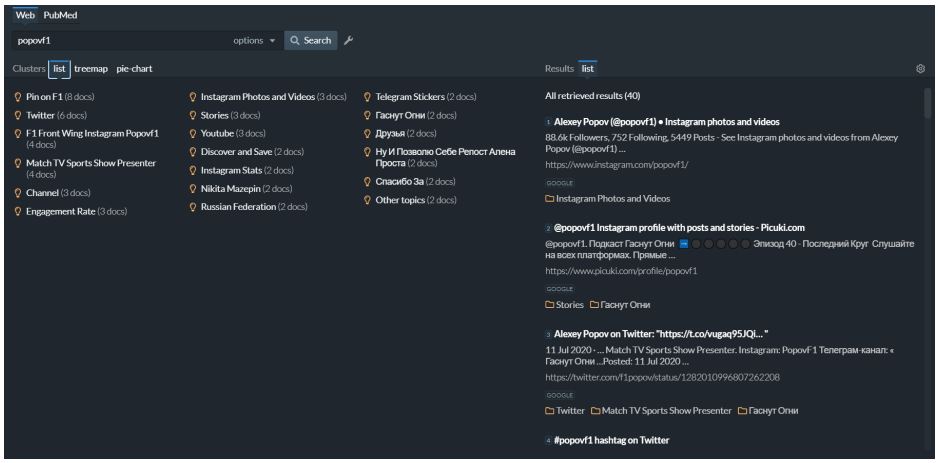

Рис. 2: Пример использования Carrot2 с разбиением результатов на группы.

3.1.3 Yippy

Yippy[2] – это метапоисковый движок, который группирует результаты поиска на категориям в группы. Наделен обширным функционалом: позволяет искать по ключевым словам новости, вакансии, правительственную информацию и блоги. Также позволяет вручную настраивать источники данных для собственного уникального метапоиска. [4]

Преимущества:

- группирует данные по тематическим категориям;
- есть возможность поиска не только ссылок в web-пространстве, но и непосредственно новостей, изображений и видео;

Недостатки:

- сервис недоступен на территории РФ;
- нет поддержки GDQ.

[2]http://yippy.com/

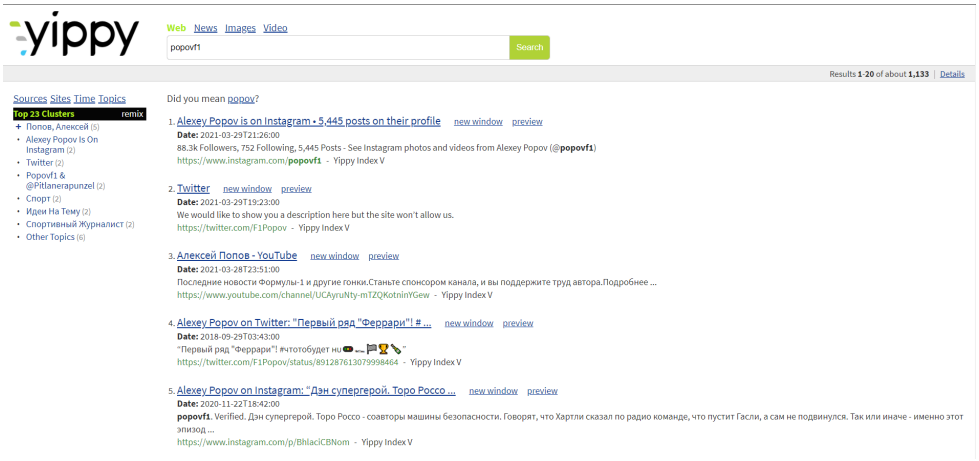

Рис. 3: Пример использования Yippy.

3.2 Поиск данных в социальных сетях

3.2.1 Maltego

Maltego[3] – это комплексное решение с множеством поддерживаемых источников информации. Представляет из себя не движок, способный просто находить ссылки и группировать их, а проводит полноценный поиск и анализ данных, выстраивает деревья взамосвязей. Например, может показать все активные адреса электронной почты заданного пользователя. [5]

Преимущества:

- выстраивание связей между объектами поиска, которыми могут быть как человек, так и группа лиц, компании, веб-сайты, организации и тому подобное;
- user-friendly интерфейс;
- возможность сохранения данных на стороне клиента с помощью СУБД;
- обладает гибкими настройками;
- является ПО с открытым исходным кодом, базовая версия которой поставляется абсолютно бесплатно в Kali Linux.

Недостатки:

[3]https://www.maltego.com/

- для доступа ко всем возможностям программы необходимо оплачивать лицензию.

3.2.2 ITools

iTools[4] – это некий агрегатор всех инструментов, перечисленных выше. Имеет возможности искать по ключевым словам людей и организаций во многих популярных современных социальных сетях. Для каждого из подключенного метода поиска имеет свои настройки.

Преимущества:

- большой перечень источников информации с настройками для каждого из них.

Недостатки:

- нет никакой аналитики и сбора данных, просто поиск и ничего более;
- нет возможности запустить сбор по всем источникам одновременно;
- данные не собираются, не хранятся. Как следствие для полноценного использования необходимо будет писать ПО поверх данного сервиса;

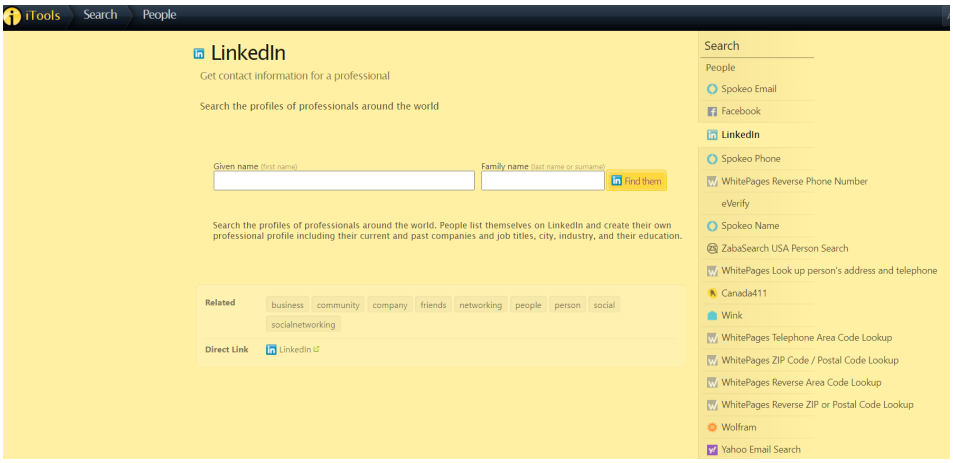

Рис. 4: Интерфейс агрегатора iTools.

[4]http://itools.com/search/people-search

3.2.3 FindThatLead

FindThatLead[5] – это онлайн-сервис, позволяющий осуществлять поиск e-mail адресов и страниц пользователей в социальных сетях LinkedIn и Twitter. Обладает возможностью проверять валидость найденного адреса электронной почты. Главным отличием является то, что можно установить данное ПО как расширение браузера Chrome.

Преимущества:

- лаконичный и понятный интерфейс, наличие расширения для браузера;
- поиск e-mail адресов по профилю в социальных сетях.

Недостатки:

- анализ данных можно совершить только вручную;
- малое количество собираемой информации;
- не подходит для комплексного и обширного анализа сущностей.

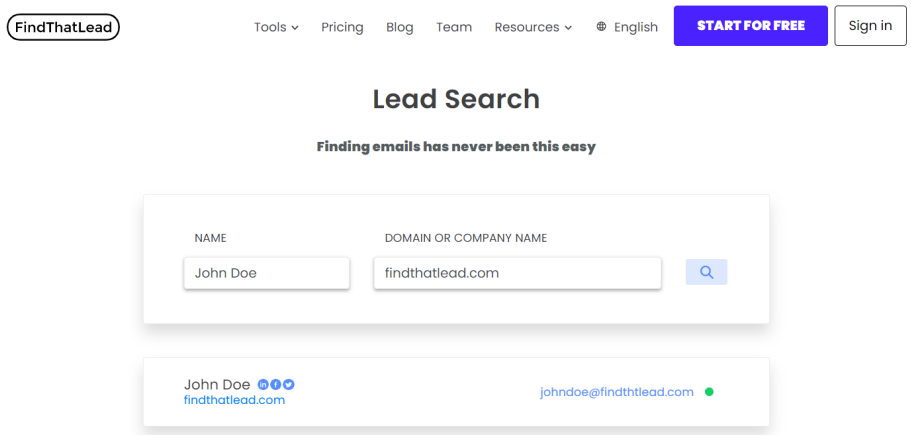

Рис. 5: Интерфейс FindThatLead.

[5]https://findthatlead.com/en

3.3 Универсальные приложения

3.3.1 Виток OSINT

Виток OSINT[6] – это отечественное решение для спецслужб, позволяет собирать информацию с помощью поисковых сервисов, так и анализируя данные социальных сетей. Строит деревья зависимостей между объектами поиска, которыми могут быть: человек, организация, событие. Имеет индексацию и дедупликацию данных, в следствие чего система не перегружена излишками данных и повышает проиводительность. Вся информация также имеет привязку к географическому положению, что позволяет более наглядно воспринимать собранные и проанализированные ПО данные.

Главным и единственным недостатком является приватность системы, программы нет в свободном доступе и оценить ее возможности вживую не представляется возможным.

3.3.2 Palantir

Palantir[7] – это зарубежное решение для спецслужб, делающее ставку прежде всего на безопасность собранной информации, удобную и развернутую подачу последней. Присутствует возможность как просто получать информацию из социальных сетей и прочих открытых источников, так и наблюдать за видеопотоком с камер наблюдений. Имеет визуализацию на карте мира.

Главным и единственным недостатком является приватность системы, программы нет в свободном доступе и оценить ее возможности вживую не представляется возможным.

3.4 Выводы

В результате исследования существующих методов сбора информации были выделены два подхода: поиск с помощью поисковых сервисов; поиск внутри социальных сетей. Однако большинство решений, которые произвоидили поиск через поисковые сервисы, зачастую не могли предоставить полноценный сбор и анализ данных, которые можно было б в последствии загрузить в СУБД для отображения в каком-либо интерфейсе.

[6] https://norsi-trans.ru/catalog/vitok-osint/

[7] https://www.palantir.com/solutions/intelligence/

Пожалуй, это главный недостаток приложений с таким подходом. Второй путь, поиск внутри соц сетей – зачастую реализован только в коммерческих проектах, и проверить объем извлекаемых данных невозможно.

4 Исследование и построение решения задачи

С целью исследования и разработки своих собственных OSINT методов сбора информации о человеке с помощью поисковых сервисов и социальных сетей предстоит решить следующие задачи:

- поисковые сервисы:
    1. Определить структуру поискового сайта. В качестве таких сайтов возьмем следующие ресурсы:
        - DuckDuckGo;
        - Google;
        - Yandex;
        - Yahoo.
    2. Извлечение найденных ссылок по заданному ключевому слову.
    3. Сбор информации с сайтов по отобранным ссылкам.
    4. Для случая с Google попробовать Google Search API: определить шаблон GET-запроса, структуру возвращаемых данных.
- социальные сети:
    1. Определить структуру сайта социальной сети. Будем работать над социальной сетью LinkedIn.
    2. Реализовать поиск и сбор данных пользователей и организаций посредством веб-краулинга сайта.
    3. Реализовать сбор данных пользователей и организаций посредством закрытого API LinkedIn. Для этого потребуется:
        - реализовать вход систему через закрытое API посредством GET и POST запросов;
        - определить шаблон GET-запроса для получения данных по указанным ключевым словам, структуру возвращаемых данных.
- реализовать все указанные выше подзадачи в систему сбора данных.

4.1 Исследование архитектуры сборщиков Scrapy

Поскольку основным фреймворком для сбора данных является Scrapy, то необходимо изначально ознакомиться с его архитектурой. [6]

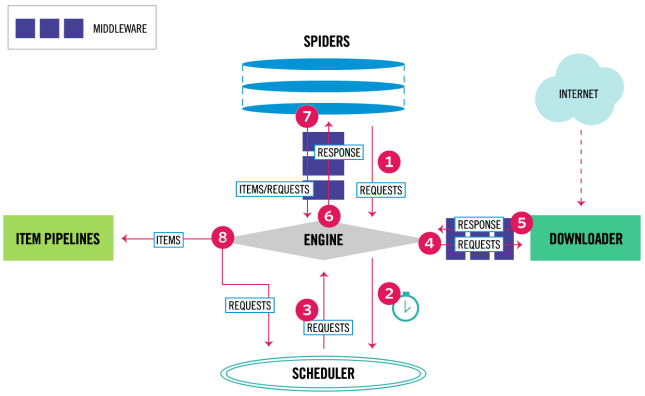

Рис. 6: Архитектура Scrapy spider.

Из рисунка видно, что изначально из spider'ов запросы направляются в движок и планировщик, затем через промежуточный загрузчик запросы выполяются в сети Интернет. Ответ от ресурсов возвращается в загрузчик, оттуда обратно в движок и в конвейер элементов. В нашей задаче потребуется писать собственные downloader middlewares и pipelines, помимо самих spiders непосредственно.

4.1.1 Scrapy Downloader Middleware

Обусловленно это тем, что на этапе, когда запрос находится в загрузчике, есть возможность загрузить api-токен, логин и пароль, или cookie-файлы для браузера и подставить его в запрос (переопределяемый метод process_request). В случае, если учетных данных нет, то в загрузчике можно составить несколько вспомогательных запросов, которые нагенерируют новые cookie-файлы и подставят в исходный запрос. Также есть возможность обработать ответ в методе process_response. Этот метод может использоваться для обновления учетных данных, кодов ответа, отличных от 200, но которые допустимы для запроса. Стоит отметить, что каждый запрос, запущенный внутри проекта, будет проходить через process_request и process_response. Это образует некую рекурсию и сложности для понимания, от какого именно запроса мы получили ответ.

4.1.2 Scrapy Item Pipelines

Изначально Scrapy просто собирает данные в некий массив структур, который можно выгрузить в json файл. Но фреймворк также поддерживает функционал скачивания файлов по их url. Для изображений используется встроенный ImagePipeline, для файлов – FilesPipeline. Но есть потребность иногда рендерить веб-страницы полностью, так как Scrapy не поддерживает выполнение JavaScript файлов. Для рендера html страниц будем использовать Splash[8] и фреймворк scrapy-splash[9]. Таким образом, для выгрузки наибольшего количества данных, документов и изображений с веб-страниц будет использовать SplashRequest, который вернет текст html-страницы с всеми отработанными JavaScript-скриптами.

4.2 Извлечение информации из страниц

В то время как Scrapy используется для навигации и перехода по ссылкам, отправке запросов и получения ответов от сайтов, в то же время он не может самостоятельно структурировать полученную информацию. Инструменты для извлечения данных с веб-страницы следующие: CSS-селекторы, Xpath-селекторы [7]. С помощью них можно извлекать данные из объекта Response, который предоставляет Scrapy после выполнения запроса. Это стандартные технологии для извлечения данных с html страниц, они весьма универсальны, но имеют один перечень минусов:

- для того, чтоб их задействовать необходимо загрузить html страницу, зачастую содержащую лишние данные, перед применением надо ждать выполнения всех скриптов на сайте;
- сайты очень часто меняют свою верстку в следствие чего селекторы могут стать неактуальным (перестать работать или собирать некорректные данные);
- многие популярные сайты имеют защиту, шифрование против извлечения данных при помощи селекторов, которые искажают ключевые поля (имена классов, id атрибутов), что усложняет со временем составление и поддержку программы.

[8] https://splash.readthedocs.io/en/stable/
[9] https://github.com/scrapy-plugins/scrapy-splash

4.3 Поиск информации в поисковом портале при помоще Google API Search

Поскольку поиск и извлечение информации с помощью CSS-селекторов весьма дорогостоящее по времени занятие, было решено исследовать другие способы поиска данных в Google. Инструментом, который решал бы вопросы затрат времени стал Google API Search. Он обрабатывает ровно те же запросы и возвращает JSON-файл с результатами поиска. И поиск посредством api куда более производительное и стабильное, то есть теперь не нужно привязывать на CSS-селекторы, которые могут стать невалидными в любое время. А поскольку данное API является открытым, имеет документацию, то и разбротка и поддержкой сборщиков становится куда более простой задачей.

4.4 Поиск информации в социальной сети посредством закрытого LinkedIn API

Социльная сеть LinkedIn широко распространена по всему миру. Однако фронтчасть сервиса также шифруется и подвергается изменениям со временем. Ровно с такими проблема и пришлось столкнуться во время разработки сборщиков, использующий CSS-селекторы. Во избежание повторения ситуации, когда весь сбор одномоментно становится сломанным и невалидным было решено найти метод, который стабильно извлекал информацию, но не зависел от изменений интерфейса. В итоге было найден один из таких вариантов[10].

Исходя из анализа, были выявлены следующие шаблоны url, с помощью которых можно получить данные:

- `https://www.linkedin.com/voyager/api/organization/companies?decorationId=com.linkedin.voyager.deco.organization.web.WebFullCompanyMain-12&q=universalName&universalName={company_id}` – возвращает информацию по заданной организации;
- `https://www.linkedin.com/voyager/api/feed/updatesV2?count=100&q=chronFeed&start={start_index}` – возвращает некоторую пачку постов пользователя (размер не превышает 10 единиц);

[10]https://github.com/tomquirk/linkedin-api

- `https://www.linkedin.com/voyager/api/identity/profiles/{profile_id}/profileView` – возвращает данные о заданном пользователе;
- `https://www.linkedin.com/voyager/api/search/blended?{filter}` – производит поиск в зависимости от того, как составить фильтр. Для поиска по людям необходимо вставить в фильтр значение «resultType->PEOPLE», для компаний – «resultType->COMPANIES»;
- `https://www.linkedin.com/uas/authenticate` – для авторизации пользователя, под учетной записью которого будет производиться поиск и сбор.

На самом деле API поддерживает специфичный язык запросов, с помощью которого можно выводить лишь определенные поля. Но исходя из того, что возвращает API, было установлено что нет таких полей, от которых стоило б отказаться. Было выявлено, что поиск среди компаний поддерживает только поиск по ключевым словам, в то время как среди пользователей список более обширный: тип (круг) связи с пользователем; регион; текущая отрасль; текущее место работы; предыдущие места работы; языки общения; интересы и увлечения; образовательные учреждения; имя; фамилия; заголовок профиля; заголовок компании; заголовок школы.

Отдельно стоит описать метод авториации пользователя. На вход принимаются логин и пароль учтеной записи, под которой будет вестить работы. По указанному выше url отправляется запрос. В ответе будут находится так называемые «незарегистрированные cookie», среди которых будет содержаться JSESSIONID, необходимый для того, чтоб составить csrf-token для доступа через API. Получив cookie, необходимо выполнить POST запрос по тому же url, в список заговков запроса включить заголовок 'Cookie: JSESSIONID; ', а в тело запроса отправить структуру из (session_key, session_password, JSESSIONID) равный (логин пользователя, пароль, JSESSIONID из первого GET запроса). Таким образовым мы связываем cookie с аккаунтом, сохраняем их у себя в СУБД и можем беспрепятственно отправлять любые запросы в LinkedIn. Поскольку авториация пользователя по факту возвращает только актуальные куки, то имеет смысл использовать данный механизм авторизации не только для API случая, но и для web.

5 Описание практической части

5.1 Описание выбранного инструментария

Работа была написана на языке Python, основной фреймворк для сбора данных – Scrapy, так как эта библиотека позволяет гибко настраивать параметры запросов, их обработку, генерацию cookie-файлов, поддерживает множественные подключения к ресурсы, асинхронно собирает данные [8]. В качестве базы данных выступает MongoDB, поскольку она хранит данные в формате JSON-подобных документов [9].

Поскольку поиск в поисковых сервисах и поиск в социальной сети LinkedIn отличается по концепции и настройке пауков Scrapy, то они были выделены в 2 различных проекта.

5.1.1 Архитектура работы сборщиков в поисковых сервисах

Диаграмма классов приведена на рис. 7.

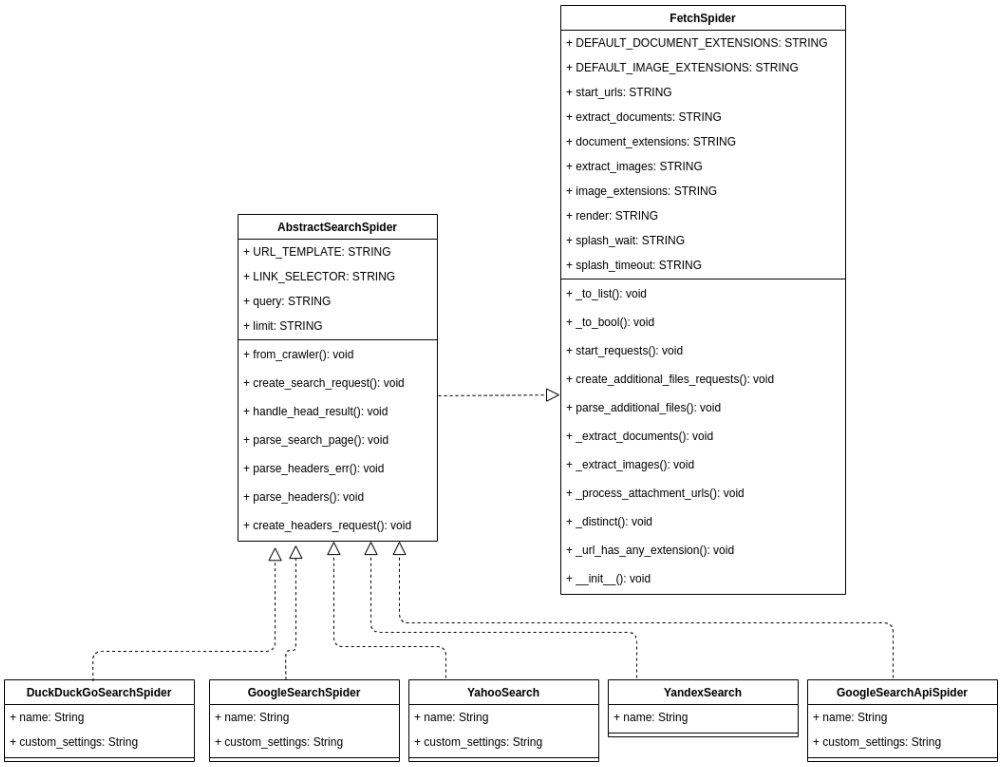

Рис. 7: Диаграмма классов сборщик в поисковых сервисах.

Система включает следующие 9 классов:

- Сборщики данных:
  - FetchSpider – позволяет собирать все документы, изображения и html-код страницы;
  - AbstractSearchSpider – содержит общие метода генерации запросов, обхода страниц и сбора данных с них;
  - DuckDuckGoSearchSpider – реализует конструктор запуска сборщика для поискового сервиса DuckDuckGo и несколько специфичных констант, таких как шаблон url с query и CSS-селектор найденных ссылок;
  - GoogleSearchSpider – реализует конструктор запуска сборщика для поискового сервиса Google и несколько специфичных констант, таких как шаблон url с query и CSS-селектор найденных ссылок;
  - YahooSearch – реализует конструктор запуска сборщика для поискового сервиса Yahoo и несколько специфичных констант, таких как шаблон url с query и CSS-селектор найденных ссылок;
  - YandexSearch – реализует конструктор запуска сборщика для поискового сервиса Yandex и несколько специфичных констант, таких как шаблон url с query и CSS-селектор найденных ссылок, настройки прокси;
  - GoogleSearchApiSpider – реализует сборщик для поискового сервиса Google, который будет производить сбор с помощью Google API Search.
- Вспомогательные классы:
  - GoogleAPICredentialsDownloaderMiddleware – данный класс производит неким проводником между Scrapy Engine и GoogleSearchApiSpider, в нем идет выбор API-ключа по стратегии «выбери тот ключ, у которого осталось наибольшее количество запросов» и обработка 429 ошибки (случай, когда API-ключ неожиданно превысил лимит использований и его необходимо признать невалидным, и запустить запрос с новым ключом);
  - SplashFilesPipeline – выкачивает все файлы, которые были получены в ходе сбора, если отобранная ссылка была ссылкой не на html-страницу.

5.1.2 Архитектура работы сборщиков в социальной сети LinkedIn

Система включает следующие n классов:

- поиск и сбор с помощью навигации по атрибутам html-кода страницы и извлечение информации из атрибутов:

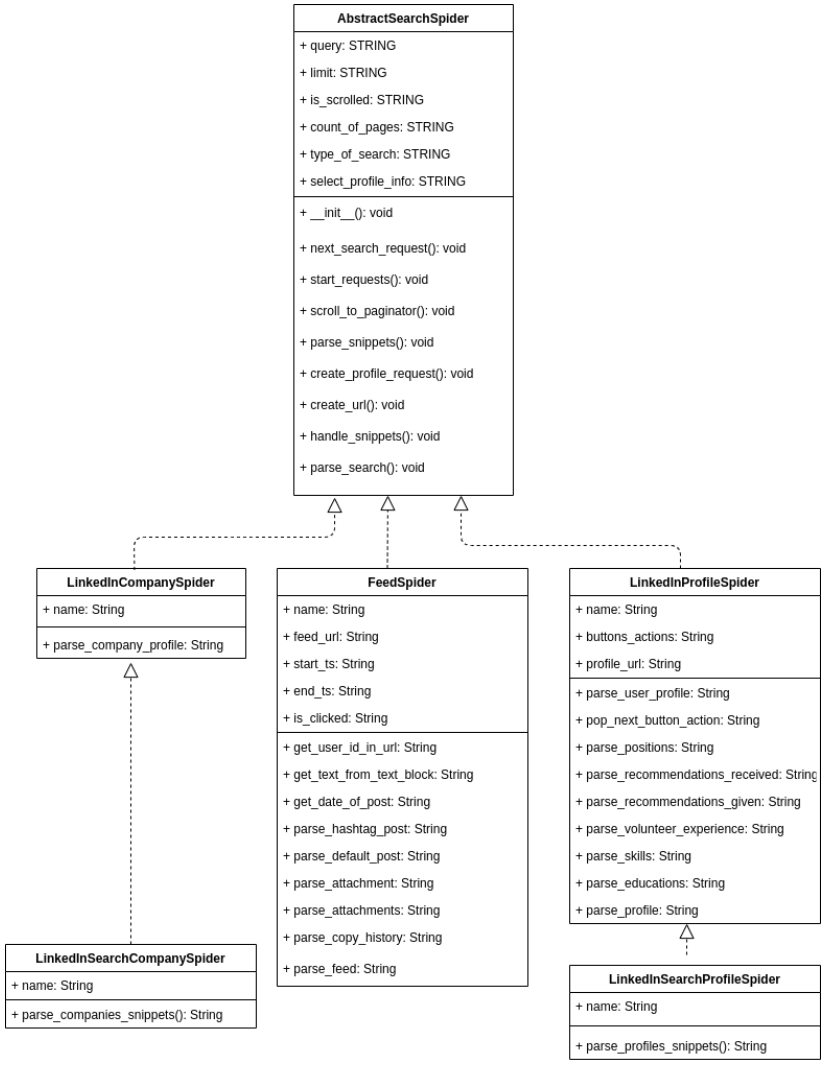

Рис. 8: Диаграмма классов сборщик в социальной сети LinkedIn web.

  - AbstractSearchSpider – абстрактный класс для поиска и сбора людей и организаций;
  - LinkedInCompanySpider – сборщик данных компаний;
  - FeedSpider – сбор данных новостной ленты пользователя;

– LinkedInProfileSpider – сборщик данных пользователей;

– LinkedInSearchCompanySpider – поисковик компаний внутри социальной сети. При настройке имеет возможность собирать информацию о найденных организациях;

– LinkedInSearchProfileSpider – поисковик пользователей внутри социальной сети. При настройке имеет возможность собирать информацию о найденных людях.

- поиск и сбор с помощью закрытого LinkedIn API:

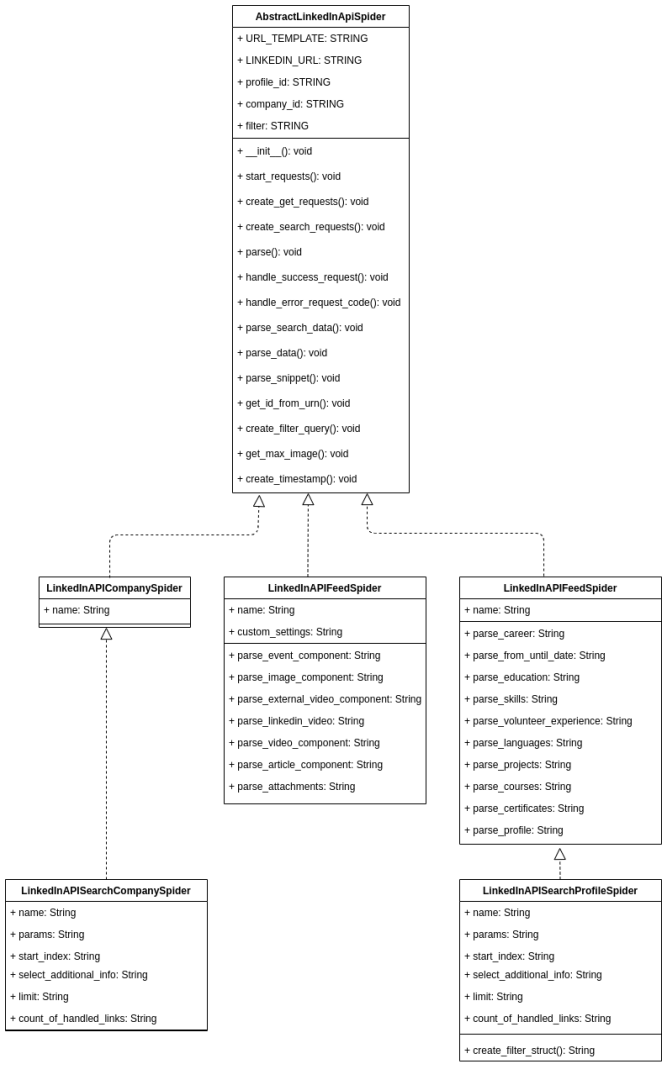

Рис. 9: Диаграмма классов сборщик в социальной сети LinkedIn API.

- AbstractLinkedInApiSpider – класс, который эмулирует для получения данных из социальной сети посредством API;
- LinkedInAPICompanySpider – сборщик данных заданной компании;
- LinkedInAPIFeedSpider – сборщик новостной ленты;
- LinkedInAPIProfileSpider – сборщик данных заданного пользователя;
- LinkedInAPISearchProfileSpider – производит поиск пользователей по заданным фильтрам. Имеет возможность собирать информацию о найденных людях при настройке;
- LinkedInAPISearchCompanySpider – производит поиск компаний по заданным фильтрам. Имеет возможность собирать информацию о найденных организациях при настройке.

- Вспомогательные классы:
  - AccountStatus – перечисление со статусом аккаунта, под которым мы пытаемся собирать информацию в социальной сети;
  - LinkedInCredentialsDownloaderMiddleware – если в приложении нет cookie файлов или имеются устаревшие cookie, данный класс перелогинивает указанный в настройках аккаунт при помощи GET и POST запросов в LinkedIn API. На выходе получаем обновленные cookie файлы и возможность дальше собирать информацию из социальной сети.

6 Заключение

В данной работе были исследованы методы OSINT для поиска информации о человеке. Её решение было разбито на следующие задачи:

- Поиск данных в поисковых сервисах:
  - Провести анализ литературы и существующих решений для извлечения данных из поисковых систем;
  - Разработать методы поиска и сбора информации из поисковых систем.
- Поиск данных в социальных сетях:
  - Провести анализ литературы и существующих решений для извлечения данных из социальных сетей;
  - Разработать методы поиска и сбора информации из социальных сетей.

В рамках решения описанных выше задач были решены следующие:

- проведен обзор литературы, статей, посвященных описанию различных OSINT-методов. Обзор показал, что методы поиска и сбора могут отличаться, самые передовые приложения самостоятельно ищут, собирают и анализируют данные, представляю их далее в виде дерева зависимостей;
- проведен обзор литературы, статей, связанных с устройством фреймворка Scrapy, Splash. Обзор показал, что на данный момент использование Scrapy полностью оправдано, если необходимо производить сбор обширных данных на протяжении большого количества времени, так и доказал, что использование Splash для рендера html-страниц полностью оправдано в нашей системе;
- Разработы методы поиска и сбора информации, собраны тестовые данные для оценки их качества;
- Проведена оценка качества с помощью сторонних метрик, которые отображают частоту вхождений каждого из полей.

По результату исследований разработаны и реализованы OSINT-методы поиска и сбора данных из поисковых источников и социальной сети LinkedIn на языке Python.

Данная работа может быть продолжена в следующих направлениях:

- проведение исследований о возможности включения большего количества поисковых ресурсов и социальных сетей в систему;
- проведение исследований о возможности построения деревьев связи и наглядного отображения зависимостей в пользовательском интерфейсе.

Следует отметить, что разрабатываемое решение для LinkedIn осуществляет переходы только по URL-ссылкам, находящимся в атрибутах. Но современные сайты могут использовать динамическую подгрузку данных, и на этот случай реализован рендер страницы при помощи Splash, а так же выявлен способ извелечения данных через API. Так что можно сделать вывод, что на данный момент система является самодостаточной: она может обрабтывать статические, динамические сайты и вызовы через API.

Список литературы

[1] *Ольга, Дзюба.* OSINT: что это, кому он нужен, какие методы сбора и типы информации использует? — 2020. — Август. https://yushchuk.livejournal.com/1451268.html.

[2] *ru.wikipedia.org.* Google hacking. — 2020. — Ноябрь. https://ru.wikipedia.org/wiki/Google_hacking.

[3] *en.wikipedia.org.* Carrot2. — 2021. — Март. https://en.wikipedia.org/wiki/Carrot2.

[4] *en.wikipedia.org.* Yippy. — 2021. — Февраль. https://en.wikipedia.org/wiki/Yippy.

[5] *Опанюк, Игорь.* Maltego. Нароет все. — 2009. — October. https://habr.com/ru/post/73306/.

[6] *Developers, Scrapy.* Architecture overview. — 2021. — Апрель. https://docs.scrapy.org/en/latest/topics/architecture.html.

[7] *Heusser, Matthew.* Selenium Tips: CSS Selectors. — 2020. — Август. https://saucelabs.com/resources/articles/selenium-tips-css-selectors.

[8] *Kouzis-Loukas, Dimitrios.* Learning Scrapy: Learn the art of effi cient web scraping and crawling with Python / Dimitrios Kouzis-Loukas. — Packt Publishing, 2016. — Pp. 198–210.

[9] MongoDB in Action, Second Edition / Kyle Banker, Peter Bakkum, Shaun Verch et al.; Ed. by Mihalis Tsoukalos. — Manning Publications, 2016. — Pp. 75–97.

[10] *https://jivoi.github.io/.* Awesome OSINT. — 2021. https://github.com/jivoi/awesome-osint.

[11] *Kozhuh.* Что такое Google Dorks? https://spy-soft.net/gugl-dorki/.

[12] *Goossens, Michel.* The LaTeX Companion / Michel Goossens, Frank Mittelbach, Alexander Samarin. — Reading, Massachusetts: Addison-Wesley, 1993.