Система управления доступом к ресурсам веб приложений на основе анализа поведения пользователя

А.А. Москаленко, О.Р. Лапонина, В.А. Сухомлин

Аннотация **– рассмотрен веб-скрапинг – процесс извлечения данных со страниц веб-сайтов в интернете с помощью автоматизации обращений к веб-сайту. В последнее время задача определения программ веб-скраперов стала актуальна. Об этом свидетельствует появление вакансий компаний, где требуется разработать средства защиты от веб-скраперов и статьи о вредоносной деятельности веб-скраперов. В статье проведено исследование поведения веб-скраперов. Выделены характерные признаки этих программ. Предлагается методика сбора и анализа данных поведения пользователей для определения веб-скраперов. Разработан модуль веб-фреймворка Django для определения программ веб-скраперов. Модуль способен собирать и анализировать данные о поведении пользователя на веб-сайте. Для сбора данных и тестирования работы модуля созданы программы веб-скраперы.**

Ключевые слова **– Веб-скрапинг, Selenium WebDriver, обход блокировок веб-скраперов, анонимность в сети Интернет, парсинг сайтов**

I. Введение

Веб-скрапинг (web-scrapping) – процесс извлечения данных со страниц веб-сайтов в интернете с помощью автоматизации обращений к веб-сайту. Копирование списка контактов из веб-каталога является примером скрапинга. Для сбора больших объемов данных необходима автоматизация, и веб-скраперы выполняют именно эту функцию. С юридической точки зрения процесс веб-скрапинга не может считаться незаконным, поскольку злоумышленник извлекает информацию, доступную всем.

Часть таких автоматизированных средств может нанести вред веб-сайтам. Во-первых, активность программ может увеличить нагрузку на веб-сервер и замедлить его работу. Во-вторых, автоматизированное извлечение данных со страниц веб-сайтов может быть нежелательно.

Цели работы. Целью данной работы является исследование методов определения программ веб-скраперов и разработка модуля веб-фреймворка Django для определения веб-скраперов. Данный модуль использует математические алгоритмы определения автоматизированной активности на сайте.

Статья получена 25 июня 2020.
А.А. Москаленко – магистр МГУ имени М.В. Ломоносова (email: alekmosk25@gmail.com).
О.Р. Лапонина – н.с. МГУ имени М.В. Ломоносова (email: laponina@oit.cmc.msu.ru).
В.А. Сухомлин – д.т.н., профессор МГУ имени М.В. Ломоносова (email: sukhomlin@mail.ru).

II. Исследование веб-скрапинга

Процесс скрапинга состоит из трех основных этапов:

- **Извлечение кода веб-страниц из интернета.** На первом этапе программа совершает HTTP запросы по URL-адресам в соответствии с логикой работы и извлекает HTML код страниц.
- **Извлечение необходимой информации из кода веб-страниц.** На этом этапе с помощью специальных механизмов (регулярные выражения, HTML-парсеры, искусственный интеллект) происходит выделение необходимой информации из HTML-кода.
- **Сохранение структурированных данных в таблицах или базах данных**

Как веб-сайты определяют веб-скрапинг

Веб-сайты определяют веб-скраперы по характерным для веб-скраперов признакам:

1. Необычное количество трафика с одного IP-адреса (например, пользователь переходит на новую страницу сайта каждую секунду сотни раз)
2. Выполнение повторяющихся задач (основано на том, что пользователь не будет выполнять одни и те же задачи много раз)
3. Использование ссылок, которые не видны обычному пользователю, но содержатся в коде веб-страницы

Способы блокировки веб-скраперов

1. Запретить доступ на сайт с определенного IP-адреса
2. Запретить идентификатор пользователя, с использованием которого злоумышленник заходит на сайт. В этом способе доступ к сайту осуществляется только авторизованными пользователями

Рекомендации для веб-скрапинга, чтобы преодолеть обнаружение

1. Настройка работы веб-скрапера

Настроить скорость работы веб-скрапера на оптимальную скорость обхода после нескольких пробных запусков и периодически проверять настройки веб-скрапера, так как сам сайт может меняться со временем. Поместить случайные программные вызовы между запросами, добавьте задержки после обхода некоторого количества страниц и регулировать количество одновременных запросов. Эти методы

делают работу веб-скрапера похожей на работу человека.

2. Использование прокси-серверов и ротация IP-адресов.

Если перед веб-скрапером стоит цель не вызывать подозрение, следует периодически менять IP-адрес. Прокси серверы с ротацией IP-адресов циклически меняют IP-адреса. Таким образом увеличивается вероятность равномерного использования всех доступных IP-адресов и снижается вероятность блокировки на сайте.

Веб-скрапер легко определить, если он регулярно отправляет похожие запросы с одного IP-адреса, если использовать ротацию IP-адресов, то обнаружение становится сложнее. Существует несколько способов изменить IP-адрес. Могут помочь сервисы VPN, прокси-серверы, технологии TOR. Кроме того, различные коммерческие провайдеры также предоставляют услуги по автоматическому изменению IP-адресов.

Идея ротации IP-адресов во время веб-скрапинга состоит в следующем: можно сделать так, чтобы повторяющиеся запросы отправлялись с различных IP-адресов. Таким образом создается впечатление, что несколько реальных пользователей совершают запросы к серверу из различных мест.

3. Изменение пользовательского агента (User Agent)

Агент пользователя - это строка, которую отправляет веб-браузер для идентификации себя, агент пользователя содержит информацию о системе и браузере от которой приходит запрос: имя браузера, версию операционной системы и другие параметры. По агенту пользователя можно определить параметры системы: название операционной системы, версию и разрядность, разрешение экрана. Можно определить какое устройство использует пользователь браузера, компьютер, телефон или планшет.

Каждый запрос, сделанный из веб-браузера, содержит заголовок агента пользователя, использование одного и того же агента пользователя приводит к обнаружению веб-скрапера. Способ обойти это обнаружение – подделывать агент пользователя и изменять его при каждом запросе на веб-сайт.

Если сайт заблокировал веб-скрапер, то можно попробовать обойти эту блокировку заменой агента пользователя или отправкой случайных записей в строке агента пользователя. Таким образом у веб-сервера создается впечатление, что к нему приходят запросы от различных браузеров, а не из одного. Стоит отметить, что рандомизация строки агента пользователя без изменения IP-адресов почти бесполезна.

III. Обзор существующих решений

В 2014 году Zhong J. разработал метод аутентификаци на основе анализа действий пользователя на сайте. Для классификации пользователей использовался алгоритм машинного обучения. В работе было проведено тестирование метода. Точность классификации превысила 90 %. [3]

В 2018 году Вишневский А. С. в своей статье [4] предложил систему для выявления хакерских атак, основанную на анализе поведения пользователей сайта.

В работе предложен способ сбора ip-адресов, с которых проводились атаки на сайт.

Таким образом у веб-сервера создается впечатление, что к нему приходят запросы от различных браузеров, а не из одного. Стоит отметить, что рандомизация строки агента пользователя без изменения IP-адресов почти бесполезна

IV. Постановка задачи

X – множество векторов признаков поведения пользователя сайта

Y – множество ответов $\{0, 1\}$. 1 – пользователь является веб-скрапером, 0 – не является

$y: X \to Y$ – неизвестное отображение из X в Y

Дано:

$\{x_1, \dots x_k\} \in X$ – обучающая выборка

$y_i = y(x_i)$ – ответ, i от 1 до k

Найти:

$f: X \to Y$ – функция, приближающая y на X

Функцию f будем оценивать по количеству ошибок первого и второго рода. Ошибка первого рода – алгоритм определил поведение обычного пользователя, как поведение веб-скрапера. Ошибка второго рода – алгоритм определил поведение веб-скрапера, как поведение обычного пользователя.

Для задачи определения веб-скраперов минимизация количества ошибок первого рода важнее минимизации количества ошибок второго рода, так как в случае ошибки первого рода будет ограничен доступ обычного пользователя к системе, это может нанести больший вред организации, чем работа веб-скрапера. Например, блокировка пользователя интернет-магазина может уменьшить прибыль компании.

Для решения задачи требуется определить функцию f. Можно выделить два подхода:

1. **Сигнатурный подход**. Основан на выявлении атак, соответствующих определенным шаблонам, например, количество запросов в минуту меньше 100. Сигнатурным методом можно легко обнаруживать известные модели поведения, но невозможно обнаружить новые модели поведения, для которых нет подходящих шаблонов.

2. **Аномальный подход**. Метод обнаружения аномалий не проверяет информацию на соответствие шаблонам. Вместо этого создаются модели надежного поведения (поведение реального пользователя) и поведение нового пользователя сравнивается с этими моделями. Для создания таких моделей предлагается использовать алгоритмы машинного обучения.

Для работы этого метода необходимо обучить систему. Сначала нужно подготовить данные, определить, какие данные характеризуют надежное

поведение, а какие характеризуют аномальное (поведение веб-скраперов).

V. РАЗРАБОТКА МОДУЛЯ ДЛЯ СБОРА ДАННЫХ

Модуль сбора данных о действиях пользователя представляет собой веб-приложение, написанное на языке Python для веб-фреймворка Django. Графическая часть приложения использует фреймворк Bootstrap4. Веб-приложение моделирует работу сайта-агрегатора, в качестве контента сайта используется информация о предложениях по работе. Модуль содержит следующие компоненты:

1. Вход на сайт

Сохраняется время входа пользователя на сайт, далее все действия пользователя ассоциируется с его логином. Для реализации входа на сайт используется встроенный механизм фреймворка LoginView из библиотеки **django.auth.contrib.view**

2. Список объектов базы данных и поиск среди этих объектов

Список объектов содержит данные, которые представляют основной интерес для программ веб-скраперов. Реализована возможность поиска среди этих объектов. Для списка объектов реализована пагинация, пользователю необходимо использовать механизм пагинации, чтобы получить доступ к следующей странице сайта.

3. Страница описания объекта

Для извлечения информации об объекте, необходимо совершить переход на страницу объекта.

Приложение будет записывать действия пользователя в файл для анализа. Будут записываться следующие действия:

1. Время входа пользователя
2. Открытие вакансий (пользователь нажал на название вакансии)
3. Нажатие на кнопку пагинации
4. Нажатие на кнопку пагинации
5. Нажатие на кнопку “Откликнуться”
6. Нажатие на кнопку поиска по вакансиям

Собранные данные преобразованы в признаки (величины вычисляются за сессию, то есть за время, которое пользователь провел на сайте)

1. Количество секунд, которое пользователь провел на сайте
2. Количество открытых вакансий
3. Количество нажатий на кнопку пагинации
4. Количество нажатий на кнопку поиска
5. Количество нажатий на кнопку “Откликнуться”
6. Среднее арифметическое время в секундах, которое пользователь провел на странице вакансии
7. Среднее арифметическое время в секундах, за которое пользователь выполнил действие
8. Максимальное количество последовательно открытых страниц

Для сбора данных о работе веб-скраперов использовались два инструмента:

1. Готовое решение - программа “Web scraper”. Эта программа представляет собой автономное расширение для браузера. Расширение позволяет создавать сценарии веб-скрапинга (извлечение данных, переходы по сайту). Веб-скрапинг с помощью этого инструмента обладает следующими недостатками:

- Не является гибким (нет возможности добавлять паузы и прочие маскирующие действия во время веб-скрапинга).
- Нет возможности изменять User-agent браузера и использовать прокси сервера

Такой веб-скрапер выполняет несколько действий в секунду и не способен выполнять действия для маскировки под обычного пользователя, из-за этого такой веб-скрапер легко определить. Таким образом необходимо разработать собственный веб-скрапер.

2. Собственный веб-скрапер на языке Python с использование фреймворка Selenium WebDriver. Веб-скрапер работает следующим образом:

1. Входит на сайт и автоматически вводит логин и пароль
2. Извлекает HTML код стартовой веб-страницы сайта
3. Выделяет информацию о вакансиях из этого кода
4. Последовательно переходит на страницы вакансий и извлекает необходимую информацию
5. Если обработаны все вакансии на странице, то программа переходит на следующую страницу с вакансиями
6. Программа заканчивает работу, когда все вакансии обработаны

Настройка работы веб-скрапера

Первое тестирование веб-скрапера показало необходимость добавление пауз между действиями

Табл. 1 Сравнение логов работы программы и человека

Логи работы веб-скрапер	Логи работы человека
2019-12-07 22:27:20;AnonymousUser;LOGIN; 2019-12-07 22:27:21;admin;VACANCY;21; 2019-12-07 22:27:21;admin;VACANCY;20;	019-12-07 22:52:23;AnonymousUser;LOGIN; 2019-12-07 22:52:25;admin;VACANCY;21; 2019-12-07 22:52:28;admin;VACANCY;20;

Логи работы пользователей преобразованы в векторы признаков.

Табл. 2 Сравнение векторов признаков работы программы (virus1) и человека (admin)

	Name	Total_time	Click_vacancy	Click_paginate	Click_search	Click_contact	Avg_time_on_vacancy	Avg_time_on_action	Consecutive_num
0	admin	76	12	2	0	0	3.083333	5.428571	5
1	virus1	10	14	3	0	0	0.500000	0.588235	12

Можно сделать несколько выводов:

1. Веб-скрапер выполняет несколько операций за секунду, в то время как человек этого сделать не может. Таким образом нужно добавлять паузы между действиями веб-скрапера, это легко сделать при помощи функции sleep() из библиотеки timer().
2. Веб-скраперам характерно открывать вакансии в порядке их следования на сайте, человеку характерно пропускать неинтересные для него вакансии. Таким образом нужно менять порядок открытия ссылок вакансий. Для этого можно использовать функцию shuffle() из библиотеки random.
3. Веб-скраперу не характерны действия, которые не способствуют извлечению данных (нажатие на кнопку поиска, нажатие на кнопку отклика на вакансии). Нужно добавлять нехарактерные веб-скраперу действия в работу программы.

На основании этих выводов разработан усовершенствованный веб-скрапер. Веб-скрапер случайным образом совершает паузы во время извлечения данных и совершает действия, которые не способствуют извлечению данных.

Измерения показывают, что можно настроить веб-скрапер таким образом, что он будет действовать похоже с реальным пользователем.

Табл. 3 Сравнение векторов признаков работы оптимизированной программы (virus2) и человека (admin)

	Name	Total_time	Click_vacancy	Click_paginate	Click_search	Click_contact	Avg_time_on_vacancy	Avg_time_on_action	Consecutive_num
0	admin	76	12	2	0	0	3.083	5.42	5
1	virus2	78	12	3	0	2	6.100	5.44	2

Для создания обучающей выборки используется усовершенствованная версия веб-скрапера. Веб-скрапер запускается 1000 раз. В зависимости от значения генератора случайных чисел выполняются операции поиска, операции нажатия на кнопку “Откликнуться”. Дополнительно собраны данные реальных пользователей сайта junjob.ru, отобрано 1000 векторов признаков активности пользователей.

VI. Описание предложенного решения

Рассмотрим алгоритм, который совмещает преимущества аномального и сигнатурного подходов. Алгоритм использует специализированную функцию расстояния для задачи определения веб-скраперов и учитывает значения расстояний между объектами.

В качестве функций расстояния использовались различные функции:

Евклидово расстояние $\rho(x, x^k) = \sqrt{\sum_1^n (x_i - x_i^k)^2}$

Квадрат евклидова расстояния $\rho(x, x^k) = \sum_1^n (x_i - x_i^k)^2$

Применяется для увеличения веса более отдаленных друг от друга элементов.

Манхэттенское расстояние $\rho(x, x^k) = \sum_1^n |(x_i - x_i^k)|$

Эта функция уменьшает влияние больших разностей (выбросов), потому что эти разности не возводятся в квадрат.

Расстояние Чебышева $\rho(x, x^k) = \sum_1^n \max(|(x_i - x_i^k)|)$

Эта функция полезна, когда нужно определить объекты как различные, если они сильно различаются по одной координате.

Расстояние на основе весов признаков $\rho(x, x^k) = \sqrt{\sum_1^n w_i * (x_i - x_i^k)^2}$; w_i - веса признаков. Вес признака вычисляется по следующему алгоритму:

1. Вычислить p_i - максимальное значение разделяющей способности признака
2. Вес признака $w_i = \frac{p_i}{\sum_{k=0}^{n} p_i}$

Был проведены сравнительный анализ алгоритмов классификации с использованием различных функций расстояния.

Табл. 4 Сравнение результатов предсказания для различных функций расстояния

Функция расстояния	Точность предсказаний
Евклидово расстояние	0.973
Манхэттенское расстояние	0.96
Расстояние Чебышева	0.972
Расстояние на основе весов признаков	0.977

Наилучшая версия алгоритма доступна по ссылке [2]. Алгоритм содержит три основных метода:

1. Этап обучения *fit()* на котором формируются кластеры и инициализируется классификатор. Этот метод поддерживает два варианта работы - **частичное обучение**, если заданы параметры *user*, *scrape*, то кластеры формируются из

данных, которые относятся к этим кластерам и **обычный метод** ближайших соседей иначе.

2. Этап предсказания (тестирования) *predict()*, на котором определяется, к какому кластеру относится объект.
3. Метод *distance()* для подсчета расстояния между объектами. В качестве параметра *metric* использовался метод подсчета расстояний на основе весов признаков.

Проведено тестирование разработанного алгоритма. Данные об активности пользователей перемешаны и разделены на обучающую и тестовую выборки в соотношении 3 к 1 соответственно. Точность предсказаний алгоритма 0.98 (отношение верно предсказанных значений к числу всех предсказаний).

Табл. 5 Результаты работы алгоритма ScraperClassifier

	y = 1	y = 0
f(x) = 1	48,7 %	1 %
f(x) = 0	1,3 %	50 %

VII. Заключение

В работе проведено исследование поведения веб-скраперов. Выделены характерные признаки этих программ. Предлагается методика сбора и анализа данных поведения пользователей для определения веб-скраперов.

Предложенное решение состоит из двух частей:

1. Веб-приложение для сбора данных об активности пользователей
2. Алгоритм для определения веб-скраперов

Предложенный специализированный алгоритм совмещает преимущества сигнатурного и аномального подходов и показывает результаты лучше, чем универсальные алгоритмы из библиотеки sklearn:

1. Точность предсказаний увеличилась на 7% (с 91% до 98%)
2. Ошибка первого рода (важный показатель для данной задачи) уменьшился в 5 раз (с 5% до 1%)

Библиография

[1] “Утечка” базы специалистов Хабр Карьеры https://habr.com/ru/company/habr_career/blog/499740/

[2] Bitbucket https://bitbucket.org/mascai/scraperclassifier/src/master/main.py

[3] Zhong J. Kind of Identity Authentication Method Based on Browsing Behaviors. Seventh International Symposium on Computational Intelligence and Design. Hangzhou. 2014. P. 279-284.

[4] Vishnevsky A. S. Content based attack detection in web-oriented honeypots. Russia. Cybersecurity issues № 3, 2018.

[5] R. Mitchell, Web Scraping with Python. USA.: O’Reilly Media, 2015.

[6] G. Hajba, Website Scraping with Python: Using BeautifulSoup. USA.: O’Reilly Media, 2018.

[7] G. Nair, Getting Started with Beautiful Soup. USA.: Packt Publishing, 2014.

[8] M. Shrenk, Webbots, spiders, and screen scrapers. USA.: Packt Publishing, 2012.

[9] J. Buelta, Python Automation Cookbook. USA.: Packt Publishing, 2018.

[10] D. Koundal Ontology Based Crawler: Semantic web application USA.: Lambert, 2013.

[11] Emilio Ferraraa,. Web data extraction, applications and techniques: A survey. Knowledge-Based Systems, Band 70, pp. 301-323., 2014.

[12] Hai Liang,. Big Data, Collection of (Social Media, Harvesting). The International Encyclopedia of Communication Research Methods., pp. 1-18., 2017.

[13] J. Hirschey, Symbiotic Relationships: Pragmatic Acceptance of Data Scraping. Berkeley Technology Law Journal, 2014.
Huan Liu, The good, the bad, and the ugly: uncovering novel research opportunities in social media mining. International Journal of Data Science and Analytics, 1(3-4), pp. 137-143., 2016.

[14] Jakob G. Thomsen, WebSelF: A Web Scraping Framework, 2015.

[15] John J. Salerno, Method and apparatus for improved web scraping. United States of America, Patentnr. 2003.

[16] G. Joyce, Data Reveals the GRAMMYs 2017 Highlights on Social Media. 2017.

[17] S. Kalvar, Is scraping and crawling to collect data illegal? USA, 2017.

[18] A. Rezai, Beware of the Spiders: Web Crawling and Screen Scraping – the Legal Position, 2017.

[19] Raulamo-Jurvanen. Using Surveys and Web-Scraping to Select Tools for Software Testing Consultancy. In: Lecture Notes in Computer Science, USA, 2016.

[20] R. Putri, Web scraping for automated water quality monitoring system. Indonesia, 2016.

[21] G. Waddell, Web Scraping and Analyzing Craigslist Rental Listings. Journal of Planning Education and Research, pp. 1-20., 2016.

[22] P. Adamuz, Development of a generic test-bed for web scraping. Barcelona, 2015.

System for managing access to web application resources based on user behavior analysis

A.A. Moskalenko, O. R. Laponina, V. A. Sukhomlin

***Abstract* - Web-scraping is a process of extracting data from web-pages on the Internet by automating web-sites requests. Importance of web-scraping is increased with developing of the Internet. This is evidenced by the appearance of vacancies in companies where it is necessary to develop protection tools against web scrapers and articles about malicious activity of web-scrapers.**

The article studies the behavior of web-scrapers. The characteristic features of these programs are highlighted. A method for collecting and analyzing user behavior data to identify web-scrapers is proposed.

The Django web-framework module has been developed for defining web-scraper programs. The module is able to collect and analyze data about user behavior on a website. Web-scrapers have been created to collect data and test the module's operation.

***Keywords* - Web-scraping, anonymity on the Internet, selenium webdriver**

REFERENCES

[1] https://habr.com/ru/company/habr_career/blog/499740/
[2] https://bitbucket.org/mascai/scraperclassifier/src/master/main.py
[3] Zhong J. Kind of Identity Authentication Method Based on Browsing Behaviors. Seventh International Symposium on Computational Intelligence and Design. Hangzhou. 2014. P. 279-284.
[4] Vishnevsky A. S. Content based attack detection in web-oriented honeypots. Russia. Cybersecurity issues № 3, 2018.
[5] R. Mitchell, Web Scraping with Python. USA.: O'Reilly Media, 2015.
[6] G. Hajba, Website Scraping with Python: Using BeautifulSoup. USA.: O'Reilly Media, 2018.
[7] G. Nair, Getting Started with Beautiful Soup. USA.: Packt Publishing, 2014.
[8] M. Shrenk, Webbots, spiders, and screen scrapers. USA.: Packt Publishing, 2012.
[9] Buelta, Python Automation Cookbook. USA.: Packt Publishing, 2018.
[10] D. Koundal Ontology Based Crawler: Semantic web application USA.: Lambert, 2013.
[11] Emilio Ferraraa,. Web data extraction, applications and techniques: A survey. Knowledge-Based Systems, Band 70, pp. 301-323., 2014.
[12] Hai Liang,. Big Data, Collection of (Social Media, Harvesting). The International Encyclopedia of Communication Research Methods., pp. 1-18., 2017.
[13] J. Hirschey, Symbiotic Relationships: Pragmatic Acceptance of Data Scraping. Berkeley Technology Law Journal, 2014.
Huan Liu, The good, the bad, and the ugly: uncovering novel research opportunities in social media mining. International Journal of Data Science and Analytics, 1(3-4), pp. 137-143., 2016.
[14] Jakob G. Thomsen, WebSelF: A Web Scraping Framework, 2015.
[15] John J. Salerno, Method and apparatus for improved web scraping. United States of America, Patentnr. 2003.
[16] G. Joyce, Data Reveals the GRAMMYs 2017 Highlights on Social Media. 2017.
[17] S. Kalvar, Is scraping and crawling to collect data illegal? USA, 2017.
[18] A. Rezai, Beware of the Spiders: Web Crawling and Screen Scraping – the Legal Position, 2017.
[19] Raulamo-Jurvanen. Using Surveys and Web-Scraping to Select Tools for Software Testing Consultancy. In: Lecture Notes in Computer Science, USA, 2016.
[20] R. Putri, Web scraping for automated water quality monitoring system. Indonesia, 2016.
[21] G. Waddell, Web Scraping and Analyzing Craigslist Rental Listings. Journal of Planning Education and Research, pp. 1-20., 2016.
[22] P. Adamuz, Development of a generic test-bed for web scraping. Barcelona, 2015.