Open Source Intelligence Telegram-bot development

KateryнаVasiuk[a], Olena Karelina[a], Valerij Muzh[a], Liliana Dzhydzhora[a]

[a] *Ternopil Ivan Puluj National Technical University 1, Ruska, 56, Ternopil, 46001, Ukraine*

Abstract

Open Source Intelligence Telegram-bot for corporate information retrieval is developed. The bot presents information about the corporate Internet domain and e-mail addresses of employees. If the e-mail address is available in data sources, the access password is displayed. Mathematical models underlying the information retrieval algorithms are considered. The software implementation of OSINT Telegram-bot in Python is described. The system testing results are given.

Keywords 1
OSINT, Telegram-bot, Python, automation, corporate information.

1. Introduction

OSINT (Open Source Intelligence) is an important direction of practical activity and scientific developments in the field of cybersecurity. There are cases when access to corporate resources does not require hacker attack, since due to ill-considered security settings, they are in the public domain. Many state information databases are publicly available; information in social networks which the users unveil themselves; compromised information resources that occurred in the deep or dark web.

The problem of retrieving information about the company or person in open sources is of great importance. Thus, it is possible to collect data on activities and their profitability, technical infrastructure and vulnerabilities of its individual components. Logins and passwords for various services can be found in the public domain. Credentials make it possible to use e-mail, bank account, cryptocurrency wallet at your discretion. Cybersecurity experts use OSINT to prevent publicly available data from hacking attacks on person or company. There are thousands of publicly available data sources, they are scattered in klirnet, darknet, telegram, discord channels, etc. Therefore, the development of OSINT automation tools is an important problem. We consider the development of OSINT Telegram-bot. The available OSINT automation tools are Recon-ng, Maltego, Spiderfoot with their own databases of information resources, used for retreiving, but not all of them correspond to the search task. OSINT has different areas: military, law enforcement, commercial, hacking, etc. We will consider how to automate the cybersecurity analyst search in public domain on the request of a company getting corporate information security services.

2. Background

Various aspects of corporate information retrieval in open sources are studied in scientific discourse. The solution for retrieving among academic information sources is developed in paper [1]. The solution for pentester`s reconnaissance automation is offered in paper [2]. Papers [3, 4], are focused on the detection of indicators of compromise (IoC). Automation of the corporate information retrieval in selected open sources is an important and unsolved problem.

ITTAP'2021: 1nd International Workshop on Information Technologies: Theoretical and Applied Problems, November 16–18, 2021, Ternopil, Ukraine
EMAIL: kateryna_vasiuk0307@tntu.edu.ua (A. 1); karelina@tntu.edu.ua (A. 2), vmuzh@tntu.edu.ua (A. 3), lilyadzhydzhora1970@gmail.com (A. 4)
ORCID: 0000-0002-6147-6438 (A. 1); 0000-0002-5628-9048 (A. 2), 0000-0002-6147-6438 (A. 3), 0000-0002-3672-4807 (A. 4)

3. Development of the requirements for Telegram-bot of OSINT automation

3.1. Functional requirements

OSINT automation project consists of three main tasks that reveal a number of performed functions. For correct operation of these functions, the requirements ensuring the result achievement should be created. The methods of this product application are shown in Fig. 1, and detailed requirements description is given in Table 1.

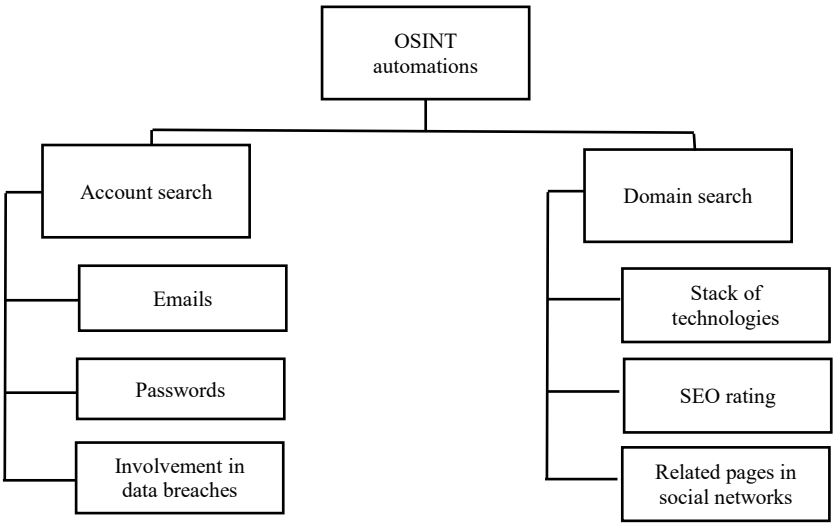

Figure 1: Methods of application of Telegram-bot for OSINT automation

Table1

Requirements description for Telegram-bot for OSINT automation according to the methods of application

Method of application	Requirements
Account search	• Recognition of the input data format • Retrieving facts of involvement in data damages using open sources • Output of available information in email@example.com:password format
Domain search	• Recognition of the input data format • Scanning of the investigation object on the open source basis

3.2. Non-functional requirements

Usability can be considered as the most important non-functional requirement. While developing the bot, the main focus is to create user-friendly interface. The following tasks should be solved:

- to implement the method of displaying information in readable format;
- to create recognition module for responding to the incorrect data input and informing the user about it;
- to ensure continuous service availability.

4. Mathematical modeling of information retrieval

Information retrieval is the process of obtaining information system resources related to information needs from these resources collection. The retrieval can be based on full-text or other indexing on the documents content basis. Information retrieval can be performed in the document, searching for documents themselves, as well as searching for metadata, searching for images, videos or sounds. Automated information retrieval systems are used to reduce information overload.

Information retrieval is the process of identifying in a certain set of documents (texts) all those that are related to the given topic, meet pre-defined conditions of the request or contain facts and data relevant to the information demand.

Information retrieval consists of four stages:

- determination (clarification) of information demand and formulation of information request;
- defining the set of possible owners of information arrays;
- obtaining information from the identified information arrays;
- awareness of obtained information and estimation of retrieval results.

The information retrieval process starts when the user inputs the query into the system. Queries are formal statements about information demands, such as a search line in the web search engines. While retrieving information, the query ambiguously identify the individual object in the collection. Instead, several objects can respond to query, perhaps with various relevance degree.

An object is an entity represented by information in the content collection or database. Depending on the application, data objects can be, for example, text documents, images, audio, mind maps or videos. The task of information retrieval, which started the development of this industry - is to search for documents that meet the query, within a certain static collection of documents. But the list of information retrieval tasks is constantly expanding and now it includes:

- information objects modeling;
- documents classification;
- documents filtering;
- documents clustering;
- designing of search engines architecture and user interfaces;
- annotation and rendering of documents.

Most information retrieval systems calculate numerical estimate how each object in the database matches the query and rank the objects according to this value. Then the user is shown the highest rated objects. Further the process can be repeated if the user wants to clarify the request.

For effective obtaining of relevant documents due to information retrieval strategies, the documents are usually converted into appropriate presentation. The search strategy includes the specific model for its submission purposes. The models are classified by two dimensions (Fig. 2): mathematical basis and model properties [5].

Types of mathematical search models:

1. Theoretical sets models represent documents as sets of words or phrases. The similarity usually follows from theoretic-set operations on these sets. The most common models are as follows:

- standard Boolean model;
- extended Boolean model;
- fuzzy presentation.

2. Algebraic models represent documents and queries in the form of vectors, matrices or tuples. The similarity of the query vector and the document vector is represented as a scalar value:

- space model vector;
- generalized vector space model;
- thematic vector space model;
- extended Boolean model;
- latent semantic indexing, i.e. latent semantic analysis.

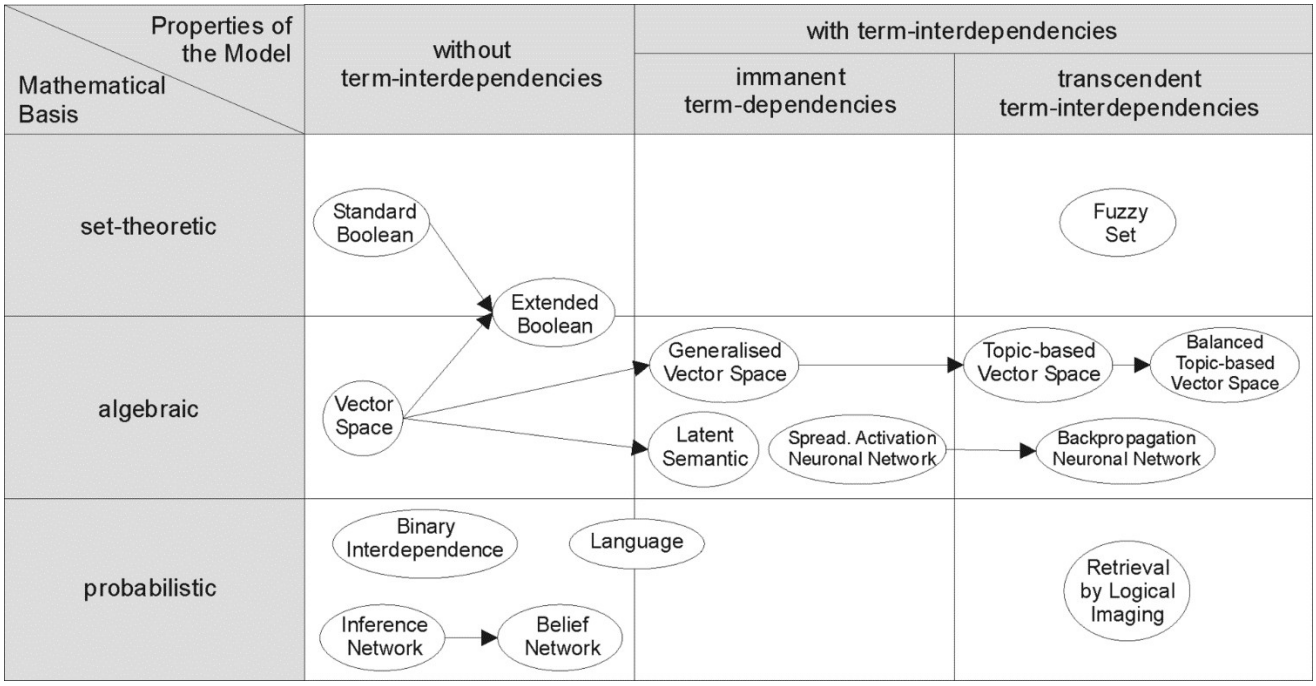

Figure 2: Classification of information retrieval models [5]

3. Probabilistic models interpret the process of searching for documents as the probabilistic conclusion. Similarities are calculated as the probability that the document is relevant to the query. Probability theorems, such as Bayesian theorem, are often used in these models:

- binary model of independence;
- uncertain conclusion;
- language models;
- model of divergence from randomness;
- hidden Dirichlet allocation.

4. Function-based search models treat documents as vectors of function values and search for the best way to combine these functions into a single relevance score, usually by ranking methods.

Models without interdependence of terms consider different terms / words as independent. This fact is usually represented in vector spatial models by the assumption of orthogonality of term vectors or in probabilistic models by the assumption of independence for term variables.

Models with transcendent interdependence of terms make it possible to present interdependencies between terms, but they do not state how interdependence between two terms is defined. They rely on the external source for the degree of interdependence of two terms (eg, human or complex algorithms).

There are many ways to evaluate how properly the found documents match the query. Precision is defined as the ratio of the number of relevant documents found by IQ to the total number of documents found:

$$\text{Precision} = \frac{|D_{rel} \cap D_{retr}|}{|D_{retr}|} \quad (1)$$

where D_{rel} is the set of relevant documents in the database, and D_{retr} is the set of documents found by the system.

Completeness: the ratio of the number of found relevant documents to the total number of relevant documents in the database:

$$\text{Recall} = \frac{|D_{rel} \cap D_{retr}|}{|D_{rel}|} \quad (2)$$

where D_{rel} is the set of relevant documents in the database, and D_{retr} is the set of documents found by the system.

Fall-out characterizes the probability of finding the irrelevant resource and is defined as the ratio of the number of found irrelevant documents to the total number of irrelevant documents in the database:

$$\text{Fall-out} = \frac{|D_{nrel} \cap D_{retr}|}{|D_{nrel}|} \quad (3)$$

where D_{nrel} is the set of irrelevant documents in the database, and D_{retr} is the set of documents found by the system.

Van Riesbergen measure.

Sometimes it is useful to combine precision and completeness in one average value. For this purpose, the arithmetic mean is not suitable, because, for example, for the search engine it is enough to return all the documents to ensure completeness equal to one unit at close to zero precision, and the arithmetic average of precision and completeness is at least 1/2. The harmonic average does not possess this disadvantage, because with the large difference in average values it is close to their minimum.

Therefore, a good measure for the joint assessment of precision and completeness is *F*-measure, which is defined as the weighted harmonic average of precision *P* and completeness *R*:

$$\mathrm{F} = \frac{1}{a\frac{1}{P}+(1-a)\frac{1}{R}}, \quad a \in [0,1]. \quad (4)$$

As a rule *F*-measure is presented in the following way

$$\mathrm{F} = \frac{(\beta^2+1)PR}{\beta^2 P+R}, \beta^2 = \frac{(1-a)}{a}, \beta^2 \in [0, \infty]. \quad (5)$$

When $a = 1/2$ or $\beta = 1$ *F*-measure gives the same weight to precision and completeness and is called balanced or F_1- measure (it is accepted to specify β value in the lower index), the expression for it is simplified:

$$F_1 = \frac{2PR}{P+R} \quad (6)$$

The use of balanced *F*-measure is optional: if $0 < \beta < 1$ precision is preferred, and if $\beta > 1$ completeness gains greater weight [6].

5. Software implementation of OSINT Telegram-bot

The implementation of the automated system will take place using the bot in Telegram messenger. Bots are third-party programs that run inside Telegram. Users can interact with bots by sending them messages, commands and built-in queries. Bots are controlled by HTTPS requests to API for bots.

At present Telegram is one of the most popular instant messaging platforms, because it enables to store messages in the cloud, not only in the device, Telegram can be used on Android, iOS, Windows and almost any other platform supporting web version.

Telegram uses its own MTProto encryption protocol. MTProto API (a.k.a. Telegram API) is API via which the telegram application communicates with the server. Telegram API is completely open, so any developer can write his/her own messenger client.

Creation of the new bot is performed using BotFather bot. Due to command/newbot you should create a new bot, as shown in Figure 3. BotFather calls for the name and username, and then creates authorization token for new bot.

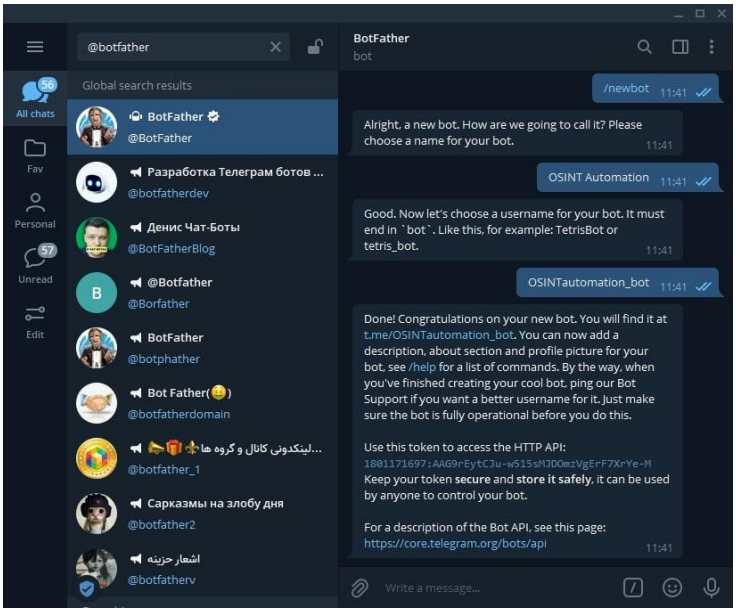

Figure 3: Creatina New bot creation

Bot name is displayed in the contact information. The username is a short name used in references and links to t.me. Usernames are 5-32 characters long and case-insensitive, and can contain only Latin characters, digital symbols, and underscores. The bot's username should end in “bot”, we called it “OSINTautomation_bot”. After bot creation, BotFather provides token. The token looks like this: 110201543: AAHdqTcvCH1vGWJxfSeofSAs0K5PALDsaw. It is due to the token that you can control the bot.

Bot design is set in BotFather: menu / mybots → Edit Bot. There you can change:

- Bot name;
- Description is the text that users will see at the beginning of the dialogue with the bot under the title "What can this bot do?";
- About is the text that will be visible in the bot profile;
- Bots Avatar, unlike users avatars and chats, cannot be animated, only pictures.
- Commands - here referred to as commands hints in the bot. More information about the commands is given below.

When the user opens the bot for the first time, he can see the “Start” button. By clicking on this button, it sends the command / start.

There are two main ways to work with Telegram in Python: by sending HTTPS requests and using Webhook. The developed project has three elemenrs: computer with Python, Telegram server and Telegram client.

Python interpreter runs on the computer, and Python program runs inside the interpreter. It is responsible for all content: it includes text templates, logic and behavior. Inside the program, Python has a library responsible for communicatioon with Telegram server. The secret key is integrated into the library so that Telegram server understands that the program is associated with the specific bot. When the client from Telegram requests information from the bot, the request is fed to the server. The request is processed by Python program, the response is sent to Telegram server, and the server responds to the client.

Bot logic is implemented by means of python-telegrambot library. This library is used for quick and simple bots creation, as it contains a large number of implemented methods of access to Telegram information.

The telegram.ext submodule is built on top of pure API implementation, it provides easy-to-use interface. The submodule consists of several classes, but two most important are telegram.ext.Updater and telegram.ext.Dispatcher. Updater class constantly receives updates from Telegram and transfers them to Dispatcher class. Then command and message processors are added to Dispatcher. They sort the updates received by Updater, and by already registered processors, the information is transferred

to the specified callback function. Each processor is the instance of any subclass of telegram.ext.Handler class. The library offers the processors classes almost for all cases.

While creating the instance of Updater class, the access token obtained during bot creation is used. The example of creating instances of Updater and Dispatcher classes, as well as the added end-user command and message processors, is shown in Listing1.

Listing 1: Bot initialization

```
from telegram.ext import CommandHandler, Filters, MessageHandler, Updater from config import BOT_API_KEY from telegram_bot_handlers import TelegramBotHandlers
updater = Updater(token=BOT_API_KEY, use_context=True) dispatcher = updater.dispatcher
start_handler = CommandHandler('start', TelegramBotHandlers.wellcome_message) help_handler = CommandHandler('help', TelegramBotHandlers.wellcome_message)
unknown_message_handler = MessageHandler(Filters.text & (~Filters.command), TelegramBo tHandlers.unknown_message) scan_email_handler = CommandHandler('scan_email', TelegramBotHandlers.scan_email) scan_domain_handler = CommandHandler('scan_domain', TelegramBotHadlers.scan_domain)
dispatcher.add_handler(start_handler) dispatcher.add_handler(help_handler) dispatcher.add_handler(unknown_message_handler)
dispatcher.add_handler(scan_email_handler) dispatcher.add_handler(scan_domain_handler)
updater.start_polling()
```

During the work, TelegramBotHadlers class containing all the functions of event handlers for the performance of the following actions is created:

- wellcome_message - displays greeting message to the user. It contains information about other available functions and their description;
- unknown_message - displays information message about incorrect data input and hint about command / help which also informs about all the functions available in the bot;
- scan_email - command to scan the email address;
- scan_domain - command to scan the domain.

The IntelxScanner () class is created in the function for e-mail addresses scanning. IntelxScanner, addresses IntelX API in order to obtain relevant data about the involvement of email address in data leaks and, if successful, provides information about the total number of data leaks and, if available, credentials, including password. The email scanning function is shown in Listing 2.

Listing 2 - Email scanning function

```
def search_email(self, email: str):
    """Method used to search email in intelx database and return results
    Args:        mail (str): String formated email [example@domain.com]
    """
    result_str = f"No information found about {email}!"
     record_count, search = self.__search_email(email)
     if record_count == 0:        return result_str
     result_str = f"Information found about {email}:\n\n"

    stats_str = self.__parse_email_stats(search=search)
     result_str = result_str + stats_str
     file_name = self.__download_first_file(search)
     email_data = self.__parse_downloaded_file(file_id=file_name, email=email)
     result_str = result_str + "\n\n" + email_data
     os.remove(f"downloads/intelx/{file_name}")
     return result_str
```

The domain scan function uses the scanner provided by BuildWith service. It provides data about the services and tools used by the domain, including their versions and descriptions, additional information such as site rankings and links to social networks, if any of them are found.

5. Testing OSINT Telegram-bot functionality

Software product testing in terms of classification by software purposes is divided into two classes: functional testing; non-functional testing. Functional testing means checking the compliance of software product with functional requirements specified in technical design specification for this product creation. Non-functional testing evaluates software product qualities such as ergonomics or performance.

During the load test, it is found that there is the requests limit to Telegram server. Bots FAQ on the Telegram website are as follows:

- no more than one message per second in one chat;
- no more than 30 messages per second in total;
- no more than 20 messages per minute in one group.

The limits can be increased for large bots by Telegram support service.

Figure 4 shows The program output with information about the data input by users, and the time of their input is shown in Fig. 4. Figure 5 shows The message output from the user is shown in Fig. 5.

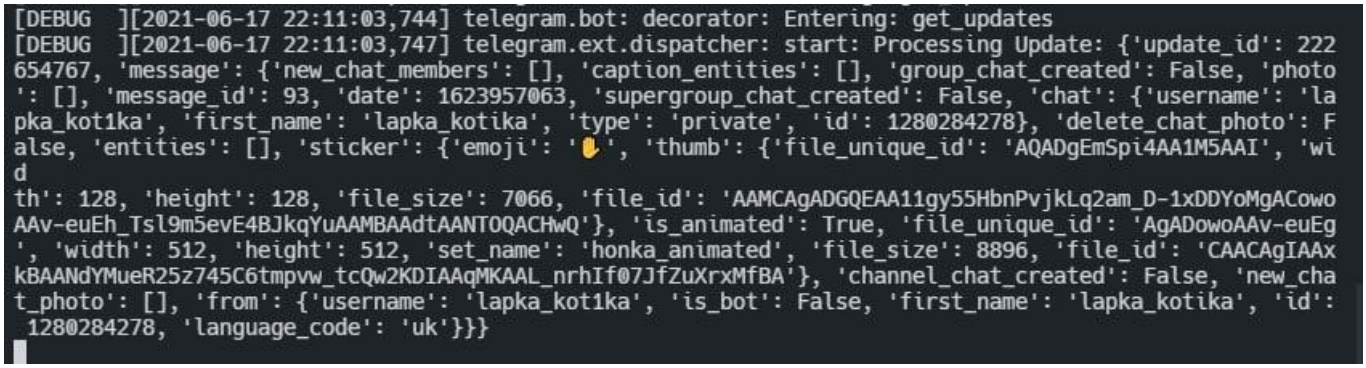

Figure 4: User message output

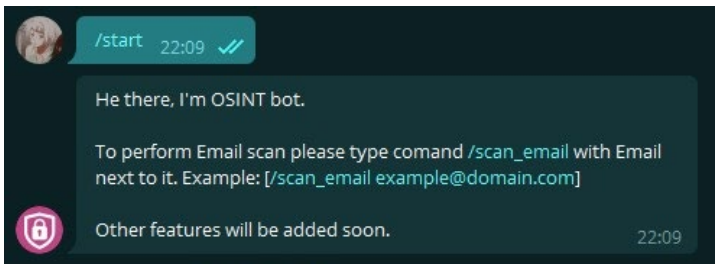

Figure 5: Greeting message input by the user

To demonstrate the successful search for the fact of data leakage concerning e-mail address, the sample, satysfying specified conditions - landry.todd@gmail.com. is selected. Figures 6 and 7 show The result of information output for an e-mail address with and without data leakage is shown in Fig. 6 and 7.

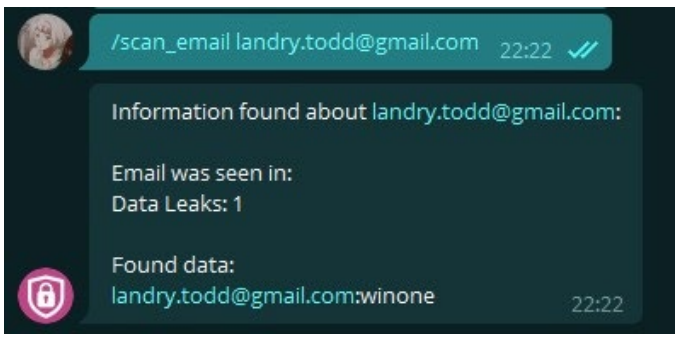

Figure 6: Checking email for data leakage with positive result

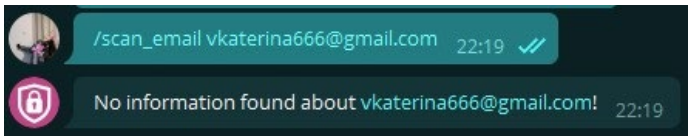

Figure 7: Checking email for data leakage with negative result

Information retrieval by domain is implemented in the bot. Due to this integration you can get detailed data about the tools used, domain rating and pages in social networks, the example of the output is shown in Fig. 8.

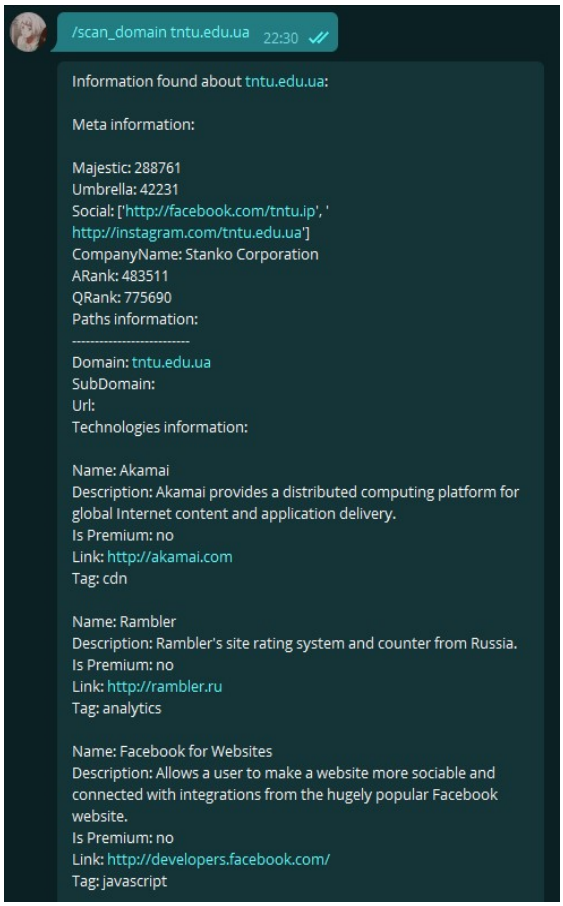

Figure 8: Output of the information about tntu.edu.ua domain

Modular and integration testing of the system components showed excellent results and did not reveal any problems.

6. Conclusions and directions of further investigations

The developed OSINT Telegram bot automates corporate information retrieval in the open sources. It is reasonable to use this solution in Security Operation Centers of cybersecurity companies. The solution is designed on the basis of the authors' experience at Cyberoo Company.

The proposed development will be expanded including new sources of information retrieval (search engines, specialized databases in Clearnet and Deep Web). For certain companies (design, design bureaus) graphic information is of primary importance. In further investigations it is necessary to study mathematical methods of similarity detection for graphic, audio, video files.

7. References

[1] B. O. Zoder. Automated Collection of Open Source Intelligence. Master`s thesis, Masaryk University, 2020, 82 p.

[2] A. Roy, L. Mejia, P. Helling and A. Olmsted, Automation of cyber-reconnaissance: A Java-based open source tool for information gathering, 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST), 2017, pp. 424-426, doi: 10.23919/ICITST.2017.8356437.

[3] R. Azevedo, I. Medeiros, A. Bessani. Automated Solution for Enrichment and Quality IoC Creation from OSINT.INForum, 2018 - researchgate.net. URL: https://www.researchgate.net/profile/Alysson-Bessani/publication/327835294_Automated_Solution_for_Enrichment_and_Quality_IoC_Creation_from_OSINT/links/5cc80ea44585156cd7bbe469/Automated-Solution-for-Enrichment-and-Quality-IoC-Creation-from-OSINT.pdf

[4] C. Martins. Generating Threat Intelligence based on OSINT and a Cyber Threat Unified Taxonomy. Ph. D. thesis, Lisbon University, 2020. – 125 p.

[5] D. Kuropka. Models for the representation of natural language documents. Ontologybased information filtering and retrieval with relational databases. Advances in Information Systems and Management Science, 2004.

[6] H. Schütze, C. Manning, P. Raghavan. Introduction to Information Retrieval. Cambridge University Press, 2008.