IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 February 2023, accepted 28 February 2023, date of publication 9 March 2023, date of current version 13 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3255165

RESEARCH ARTICLE

Exploring Dark Web Crawlers: A Systematic Literature Review of Dark Web Crawlers and Their Implementation

JESPER BERGMAN AND OLIVER B. POPOV, (Member, IEEE)
Department of Computer and Systems Sciences, Stockholm University, 114 19 Stockholm, Sweden

Corresponding author: Jesper Bergman (jesperbe@dsv.su.se)

This work was supported in part by the NordForsk Grant 80512 for the project "Police Detectives on the TOR-Network."

ABSTRACT Strong encryption algorithms and reliable anonymity routing have made cybercrime investigation more challenging. Hence, one option for law enforcement agencies (LEAs) is to search through unencrypted content on the Internet or anonymous communication networks (ACNs). The capability of automatically harvesting web content from web servers enables LEAs to collect and preserve data prone to serve as potential leads, clues, or evidence in an investigation. Although scientific studies have explored the field of web crawling soon after the inception of the web, few research studies have thoroughly scrutinised web crawling on the "dark web", or ACNs, such as I2P, IPFS, Freenet, and Tor. The current paper presents a systematic literature review (SLR) that examines the prevalence and characteristics of dark web crawlers. From a selection of 58 peer-reviewed articles mentioning crawling and the dark web, 34 remained after excluding irrelevant articles. The literature review showed that most dark web crawlers were programmed in Python, using either Selenium or Scrapy as the web scraping library. The knowledge gathered from the systematic literature review was used to develop a Tor-based web crawling model into an already existing software toolset customised for ACN-based investigations. Finally, the performance of the model was examined through a set of experiments. The results indicate that the developed crawler was successful in scraping web content from both clear and dark web pages, and scraping dark marketplaces on the Tor network. The scientific contribution of this paper entails novel knowledge concerning ACN-based web crawlers. Furthermore, it presents a model for crawling and scraping clear and dark websites for the purpose of digital investigations. The conclusions include practical implications of dark web content retrieval and archival, such as investigation clues and evidence, and related future research topics.

INDEX TERMS Cybercrime, digital forensics, systematic literature review, dark web crawling, Tor.

I. INTRODUCTION

The high level of confidentiality and limited traceability have made cybercrime on the Internet, particularly on the so-called dark networks, considerably more challenging to investigate. The tight protection of data travelling through the dark networks has rendered law enforcement with tedious and complex tasks that require extraordinary resources in terms of time, labour, knowledge, and competence.

There several different software available that execute code which connects a computer to one of the circa half-dozen dark network available today. Despite being different, they all have in common that they use state-of-the-art encryption algorithms and network traffic routing protocols that do not leave unnecessary traces. Since the trails of the traffic routed through the network are minimal, and decryption of data is not efficient nor realistically feasible, evidence must be collected elsewhere.

The largest of the dark networks, or more formally anonymous communication network (ACN), is Tor. Tor constitutes a network of servers, some of which are web servers that

The associate editor coordinating the review of this manuscript and approving it for publication was Tiago Cruz.

comprise the so-called "dark web". More correctly, they comprise one of the dark webs; other dark networks such as Lokinet, Freenet, IPFS, and I2P also include a number of servers that comprise dark webs specific to their respective network. Tor has, however, emerged as the most commonly used and essential ACN for citizens in non-democratic or semi-non-democratic countries, as well as for whistle-blowers and journalists in need of end-to-end anonymity [93].

Nevertheless, the anonymity provided by Tor is equivocal; the well-founded privacy and encryption scheme of the Onion Routing protocol is not discriminant against its users. Whistle-blowers and criminals alike benefit from the same liberating encryption algorithms and anonymous traffic routing. The unethical use of anonymity by various cybercriminals include hosting of malicious servers, illicit and illegal content, which create an arduous digital policing arena for law enforcement. To a large extent, although not exclusively, the criminal activity using or being dependent on Tor is concentrated to Tor websites textual and graphical content such as dark marketplaces, child abuse websites, hacking web fora, and akin illicit or illegal website content.

The Tor network is designed to encrypt its traffic in different layers with different keys for each layer between each server in the network, using up-to-date standardised encryption algorithms. It consists of more than eight thousand servers, or *relays*, that encrypt and route data through the Internet cables around the world. For each connection that is made through the Tor network, a minimum of three relays is required to build a circuit for anonymous Onion Routing. The first relay encrypts the data with one key, the next encrypts it with another key, and the third encrypts it with yet another key. The result is an onion like layer structure of encrypted data and encrypted encrypted data. Anonymity is upheld by the principles of the routing protocol that requires multiple relays to create a circuit; no single relay knows the complete chain of transmission.

As the possibilities of network traffic analysis and decryption are limited on ACNs, collection of web content is a profitable and fruitful alternative technique. Manual web monitoring, web intelligence gathering, and undercover operations have proven to be successful means of identifying suspects [19].

Web crawling, i.e. automated collection of web content, is an effective technique for gathering data that is unencrypted on the Tor web that excludes a lot of manual work. The web scraping technique is widely used on the clear web for commercial and utilitarian purposes. News aggregation services, price comparison services, and digital preservation units amongst national libraries worldwide use web crawling to gather, store, and archive data that is of value to the future.

Historical snapshots, or copies, of web pages or entire websites, have been included as evidence in multiple large criminal investigations in recent years. Historical screenshots of the Silk Road 2.0 occurred in the court case against the suspected operator of the notorious dark marketplace [14]. Another example is the court complaint against the suspected operator of AlphaBay, in which screenshots were also enclosed to build the case [76]. Furthermore, the evidence presented against a suspect connected to the Swedish language dark marketplace Flugsvamp 2.0 consisted of historical copies of the website scraped by the Swedish Police [33]. A noticeable amount of servers on the Tor network are reportedly volatile and disappear from the network after some time [8]. Consequently, snapshots and historical copies of websites potentially comprise unique and essential data.

Once acquired and preserved, web content can be used as traces, clues, or evidence in investigations. Moreover, it can also be further explored and dissected by examiners, investigators, and computer programs empowered with data analysis such as statistical processing, machine learning, and artificial intelligence.

Due to the volatile nature of websites, website crawling and acquisition requires a rigid and reliable programming logic to maintain and uphold the forensic scientific principles of data integrity in order for it to be admissible in a court of law. There are a number of different web crawler software available today. Some of them are customised for forensic acquisition of web content, and others are optimised for performance and breadth and coverage of large quantities of web pages. Current and previous research pertaining to web crawling has mainly focused on the clear web, and not anonymous communication networks.

The survey and literature review research of dark web crawlers is scarce to date. By further exploring this topic and identifying the characteristics of dark web crawlers, taking into account the aforementioned properties of anonymous communication networks and their discrepancy from the regular Internet and the clear web, a general understanding of the landscape of dark web crawlers can be presented to provide knowledge regarding practical dark web crawling construction and usage. Possibly, this knowledge will assist researchers and practitioners in using and developing tools for crawling websites on ACNs.

In addition to the knowledge contribution, an implementation of a Tor based crawler is presented and evaluated as a use-case in the second part of this study. The design of the crawler was based on the result from the literature review and extends an already existing dark web investigation toolkit.

A. DISPOSITION

The structure and the organisation of the paper consist of seven chapters and a bibliography. This paper presents two research contributions: one systematic literature review and one experiment-based web crawler implementation. Section wise, the first chapter is the introductory chapter that briefly describes the nature of anonymous communication networks, website content and web crawling, and how it can be used in digital investigations. Chapter two extends the scientific foundation from which the research problem and the research questions spring. Chapter three contains the systematic literature review. Chapter four includes the design science-based development and evaluation of a dark web crawler, based on the results from the literature review. Chapter five presents the results from the crawler implementation in chapter four, and chapter six discusses the results. The final chapter includes conclusions and suggested topics for future research based on the findings of the current paper. The last sections of the paper contain the bibliography and acknowledgements.

II. RELATED WORK

This section consists of an overview of previous research in the area of clear and dark web crawling and cybercrime investigation. The upcoming subsections comprise a background of knowledge that lead to the research problem and the motivation of the subject matter of this article.

A. THE WORLD WIDE WEB

Before the world wide web was established in the early 1990s, other protocols than the HTTP were present on the Internet. Gopher was one of the preceding protocols of HTTP. Gopher was a text-based protocol with focus on network file sharing. As a consequence of the entrance by graphics-enabled hyper text markup language (HTML), and rumours of licensing of the Gopher software, the world wide web became the conquering information sharing technique in 1994-1995 [24].

HTTP utilises the Internet Protocol (IP) and the Transport Control Protocol (TCP) to transmit data such as HTML pages, images or video. The procedure of retrieving an HTML page, or a web page, via HTTP is today the same as in the year 1990: a client (web browser) sends a GET request for a web page to a web server and gets a response in return. If the page is available the HTTP code 200 is sent, if the page was not found, a 404 code is sent. There are multiple other response codes specified in the HTTP standard [37], although 200 or not 200 is the essential response for most web crawlers and web users.

The HTTP standard has been revised since 1989, from version 0.9, to versions 1.1, and 2.0, up to the latest version, HTTP 3 [37], [38]. However, all HTTP versions support backward compatibility, which means that all requests and responses are the same as in HTTP version 1.0 [36], [37], [38].

Automating the GET requests to a website and following URLs found on it and sending GET requests for them, and storing the responses, is in simple terms what is known as web crawling. Web crawlers can be based on web browsers, or modified versions of web browsers, that automatically send HTTP GET requests. They can also be of smaller size in the form of software programmed to communicate using HTTP. As of today, there are many HTTP communication software libraries available for different programming languages, such as Haskell,[1] Lisp,[2] Go[3] simplifying the process of creating HTTP-based clients and servers.

B. WEBSITE ACQUISITION TOOLS

There are both open and closed source tools available for forensic acquisition of websites - i.e. websites that have been saved, or *scraped*, according to the principles of forensic science. A few of these website acquisition tools, support web crawling, and some support dark web crawling, albeit they were not designed to be powerful web crawlers.

OSIRT is a web browser tailored for non-technologically savvy investigators that supports dark websites (Tor) as well as clear websites [66]. Reportedly OSIRT is widely used by law enforcement in the United Kingdom and supports video capture and video recording of website acquisition, as well as screenshots and audit log files to uphold the chain of custody in the investigation process [92].

FAW is a proprietary website forensic acquisition tool, shaped as a web browser, that is used by Police departments around the world to acquire web content from websites, social networks, and dark (Tor) websites. FAW also supports website crawling [56].

Hunchly is another proprietary tool in the form of a web browser add-on that is built for both clear- and dark (Tor) website acquisition. Hunchly is designed to fit the needs of law enforcement [35].

C. WEB CRAWLERS

The world wide web was invented in 1989-1990 [91], and one of the first scientific articles relating to web crawling was published in 1996. By using a web crawler called Inktomi, researchers could successfully collect circa 2.6 million web pages as of November 1995 [94].

The HTTP has not changed in the way web pages are requested or served, and the technique for web crawling is the same today as it was in its infancy. There are ample web crawlers available today. The three most common crawlers according to Kumar, Bhatia, and Rattan [49] include Nutch,[4] Crawler4j,[5] and Mercator.

Performance-wise, a research study by Yang and Thieng-buranathum [97] showed that Heritrix[6] was one of the most scalable web crawlers available. In terms of robustness, the author states that the Python-based crawler library Scrapy was one of the most prominent ones.

Clear web crawlers such as the above mentioned could potentially be used when a Tor socket or Tor proxy is put in front of it; thus an effective, i.e. functional, Tor crawler could be any clear web crawler used in combination with a local Tor socket connection. Although, there might be a risk of DNS

[1] https://hackage.haskell.org/package/HTTP
[2] https://github.com/fukamachi/fast-http
[3] https://pkg.go.dev/net/http
[4] https://nutch.apache.org/
[5] https://github.com/yasserg/crawler4j
[6] https://github.com/internetarchive/heritrix3

request leaks and other privacy dire straits in case the configuration to the Tor network is not set up properly [89], [90].

D. THE TOR NETWORK AND THE TOR WEB

The regular web, or the clear web, uses, as mentioned, TCP and IP to transmit HTTP requests. Anonymous communication networks, or dark networks, use TCP and IP to transmit their own anonymous protocols. Tor transmits its onion routing (OR) protocol using the aforementioned transport protocols, likewise is I2P's garlic routing (GR) protocol escorted by them. On top of the OR or the GR protocols, ACNs convey the HTTP to, for example, serve web pages.

Although IP addresses are used to establish connection between all relays (nodes) in the Tor network, there are no IP addresses to Onion Services (previously known as Hidden Services) of which some host websites that is part of the ''dark web''. Onion Services are accessible only if their URL is known, for example http://juhanurmihxlp77nkq76byazcldy2hlmovfu2epvl5ankdibsot4csyd.onion is publicly known to be the search engine Ahmia's Onion Service [65]. There is no possibility of scanning a closed space of IP addresses to find an Onion Service on the Tor network, as could be done on the regular Internet, neither would it be an effective approach to try to pseudo-randomly guess Onion Service URLs, although it is theoretically possible [21].

Tor websites, also known as ''onionsites'', do not differ from clear web pages; the appearance is similar, they are constructed in the same way with text, images, HTML, CSS, JavaScript and they are also transferred using HTTP. The content of onionsites, however, tend to differ from the clear website due to their nature of being located on an anonymous communication network. Onionsites usually prioritise confidentiality, privacy, and anonymity over usability and performance. Therefore JavaScript is seldom implemented on these sites due to the risk of revealing the Tor user's real identity by using it [59], [75].

E. DARK WEB CRAWLERS

The aforementioned clear web crawlers, such as Nutch or Mercator, are capable of crawling ACNs like Tor. However, they require modifications to proxy a connection to the Tor network and accomplish a crawling task. However, the characteristics of the clear web and the dark web differ. Thus, contributions have been made in academia to develop time efficient, powerful, and adequate dark web crawlers for acquiring and analysing web content from Tor and other ACNs.

Hayes, Cappa, and Cardon [31] proposed a Tor web crawler capable of scraping vendor accounts from a dark marketplace and then plotting a link graph based on the vendor accounts data to investigate possible criminal activity [30].

Research has also been focused on automating the cybercrime web content collection process on both the clear and the dark web. Zulkarnine et al. [103] extended an already existing child exploitation identification crawler developed by Bouchard, Joffres, and Frank [9] to crawl both Tor and non-Tor websites simultaneously with the objective of identifying extremist and terrorism content. In summary, the crawler managed to retrieve 260 GB of data from roughly 54.000 Tor web pages [103].

Multiple research studies have approached the same problem in a similar manner, namely by developing dark web crawlers for harvesting and computationally analysing web content; these include [5], [13], [39], [73], and [41], [83].

Despite the vast number of research articles in the area, there are remaining challenges for dark web crawlers. Dark marketplaces are often protected by CAPTACHAs and authentication mechanisms that obstruct crawlers and need to be bypassed in an effective manner. Moreover, some sites implement ''crawler traps'' to hinder web crawling robots from harvesting the content from the server, such as infinite loops of web pages that do not exist or automatically pseudo-randomised pages and links that the crawler endlessly follows and downloads, exhausting it with nonsense [18].

F. RESEARCH MOTIVATION

To date, a thorough literature review of dark web crawlers, like there are for clear web crawlers, is missing. Anonymous communication networks are designed and operate in a different manner than the clear web and the regular Internet. Therefore, crawlers need to be programmed and configured accordingly to complete their crawling tasks successfully. As pointed out in previous sections, there are dark web crawlers available that have been developed by both private and public actors; however, no scientific study has systematically reviewed them. One of the objectives of this research was thus to present a rigorous assessment of existing dark web crawlers developed or used in scientific literature. The second objective was to implement the dark web crawler most frequently used in academic research to fit it into an existing toolset lacking a verified and comprehensive crawler and evaluate its performance.

III. SYSTEMATIC LITERATURE REVIEW

The first step, (1) planning, includes the preparation of the SLR: sketching out the background of the research, the research question, study selection and study quality assessment criteria, as well as data extraction and dissemination strategy.

Phase two (2), conducting a literature review, on the other hand, is more extensive and presented in further detail as follows. Phase two includes the activities: (1) study selection, (2) study quality assessment, (3) data extraction, and (4) data synthesis. Each activity is presented in the upcoming four sections.

The third phase, (3) reporting, includes specifying the dissemination mechanism as well as the formatting and evaluation of the report. These were implicit due to the nature of this report, which essentially is a peer-reviewed research article that undergoes evaluation and is publicly disseminated.

A. RESEARCH QUESTIONS

All of these activities were carried out as instructed, and the remaining concrete outcome was the research questions that were specific to the SLR (i.e. this segment of the article), which is not equal to the research questions of the entire article:

1) Which crawlers and/or scrapers have been used in scientific literature to collect data from the Tor network?
2) How do crawlers and/or scrapers used to collect data from the Tor network route the traffic?
3) Which programming languages and libraries have been the most common for programming crawlers and/or scrapers on the Tor network?

B. SEARCH STRATEGY

The documents to be selected for review are referred to as "studies" by Kitchenham [45]. In this work, the research database that was utilised for retrieving studies was Scopus.[7] The reason for choosing only Scopus as the primary resource was its complete coverage of scientific research articles, in combination with its powerful API for searching, finding, and fetching articles and their META data.

Scopus includes articles dating back to the late 18th century and indexes from the databases ACM digital library, ScienceDirect, SpringerLink, and IEEEXplore [84]. Due to the young age of the Tor network and the research of the "dark web", the risk of missing any historical articles was considered low. The search queries did, therefore, not have any publication date preference.

A Python script was developed to fetch articles from Scopus. The script can be found on https://gitea.dsv.su.se/jebe8883/SLR

C. STUDY SELECTION STRATEGY

A total number of 59 articles were retrieved from the database using the keywords `TITLE-ABS-KEY ((dark AND web AND crawler) OR (dark AND web AND scraper) OR (tor AND crawler) OR (tor AND scraper)) AND LANGUAGE(english)` where TITLE-ABS-KEY, i.e. title, abstract, and keywords define the META data in which the specified terms were searched. "LANG" specifies the language English; non-English articles were excluded from the search results.

Once the articles had been found and fetched by the script, they were written to disk with their META data, i.e. title, abstract, keywords, authors, and DOI, as seen in the example below. This sequence made data processing more manageable than working with the web-based service. In addition, the search and selection process remains more transparent when publishing the source code of the script that performed it.

```
Title: Implementing UTM based on PfSense
platform
Abstract: Today, as~Network environments
become more complex and cyber and Network
threats increase\ldots
Authors: Asghari V.
Publication: Conference Proceedings of
\ldots
Publication Type: Conference Proceeding
Article Type: Conference Paper
Scopus ID: SCOPUS_ID:84971439968
DOI: 10.1109/KBEI.2015.7436210
URL: https://api.elsevier.com/content/
abstract/scopus_id/84971439968
Keywords: Keyword1, keyword2
Time: 2022-06-01 14:22:13.125470
```

1) INCLUSION AND EXCLUSION CRITERIA

According to Kitchenham [45], different criteria for including certain articles and excluding others are crucial for initially identifying studies that relate to the research question. Naturally, a number of articles were excluded already in the database search, which only included English language articles relating to the search terms specified in the previous section. In this section, further inclusion and exclusion of articles are explained.

In this systematic literature review, the focus was on content crawling and scraping on the Tor network. However, articles that do not explicitly concern Tor were decided to be included, given that they relate to, or mention, the impact the article might have on the Tor network. The reason for this inclusion scheme was to limit the risk of missing relevant or semi-relevant articles from the selection process.

- Inclusion criteria:
  - Articles that concerned crawling, scraping, intelligence gathering, or monitoring of servers on the Tor network.
  - Articles in which a crawler or scraper was used to retrieve data from the Tor network.
- Exclusion criteria:
  - Articles that did not mention the Tor network in regards to crawling or scraping.
  - Articles that did not concern any sort of content retrieval from remote servers (on the Tor network).
  - Articles that were not peer-reviewed research articles, i.e. journal articles, conference proceedings, workshop proceedings.

The low number of articles from the search query enabled a manual assessment to be made. As a first inclusion or exclusion assessment, the abstracts for all 59 articles were manually inspected and excluded or included based on the previously specified criteria. By inspecting the titles of all articles, it appeared that [20] and [18] had the exact same title. The former was a conference proceeding article comprised of nine pages, and the latter was a journal article of fourteen pages. The conference article was filtered out since it was considered a briefer version of the journal article.

[7] https://scopus.com

The same action was taken for the conference and journal articles [39] respectively [40], where the latter was favoured. Similarly, there was a similar pair of the conference article [72] with the same title and DOI as the journal article [72]. The conference article was excluded in favour of the journal article.

The total number of articles that remained after removing conference and journal duplicates was 56.

D. STUDY QUALITY ASSESSMENT

After the initial sieving process, the guidelines by Kitchenham [45] suggest a quality assessment is done to filter out any possibly indecent studies based on a set of quality assurance checkpoints.

The remaining 56 articles were quality-checked for: bias, inconsistencies, and validity, in addition to the established inclusion and exclusion criteria.

1) EXCLUDED ARTICLES

Out of the 56 remaining articles, 15 were excluded based on the exclusion criteria after reading their abstracts. These excluded articles are recorded in Table 1.

2) INCLUDED ARTICLES

After subtracting the excluded articles, 41 articles remained for the quality assessment. The included articles and summary of them, as well as an external link to their source code of software repository, can be found in Table 2.

E. DATA EXTRACTION STRATEGY

The 41 documents that remained after the quality assessment were extracted for further analysis and assessment. In the META data from each of the articles fetched from Scopus, there was a link to the full article. These articles were manually downloaded, and then data were extracted from them.

The included and relevant articles are presented in Table 2. The ACN-based web crawler or scraper used in each of the articles is presented in the table. This table also includes a link to the source code of the crawler/scraper used, given that it is open-source software and the code is publicly available.

However, during the data extraction phase, seven articles not relevant were discovered and excluded from the review. These articles concerned crawling, but not on the dark web, as was the case with [53] and [4], or articles using a different definition of the dark web - i.e. one that does not mean anonymous communication network, such as I2P, Freenet, Lokinet, as in [51], [98] and [28]. Additionally, studies that used the Tor network and crawlers separately, as in [70], were excluded. A summary of the articles excluded during the data extraction can be found in Table 3.

F. DATA SYNTHESIS STRATEGY

Synthesising the data collected and analysed in the SLR is the final step in the process, according to Kitchenham and Charters [46]. The data synthesis strategy applied in this SLR was descriptive content-focused, where the interest was in web crawlers used in the selected studies.

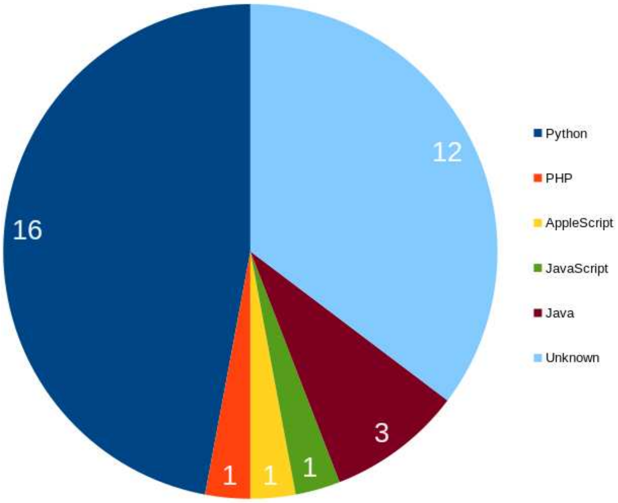


FIGURE 1. Programming languages used in articles found in the systematic literature review.

A total of 35 relevant studies were collated and summarised in the synthesis stage. It can be concluded that a minority of them used or promoted the source code of the crawlers publicly; six out of 35 did so.

The most common way of collecting dark web content in the scrutinised articles was to build a custom crawler specifically for that purpose. The most common programming language used was Python, and the most common crawler libraries used were Selenium and Scrapy. The pre-existing crawler that was used in more than one study was Apache Nutch, which was used in two studies: [39] and [42]. It should be noted that in both aforementioned studies, Nutch was customised to fit the study design and not used out of the box. In addition, the two studies using Nutch as a crawler were written by the same five authors, with two more authors in [42] than in [39].

- The most commonly used crawler was Apache Nutch. It was used in two out of 34 relevant studies.
- The most common programming language was Python, which was used in 16 out of 34 articles, see pie chart in Figure 1.
- The most common crawler library mentioned in the selected articles was Selenium. It was used in six out of 34 articles.
- The next most common crawler library mentioned was Scrapy, which was used in four out of 34 articles.

IV. IMPLEMENTATION OF A TOR CRAWLER

The second part of this article complements the systematic literature review with an implementation based on the results from the systematic literature review in the previous section.

TABLE 1. Articles excluded from the initial exclusion phase in the systematic literature review.

Title	Reference	Reason for Exclusion
InTIME: A machine learning-based framework for gathering and leveraging web data to cyber-threat intelligence	[48]	No crawler used.
Link Harvesting on the Dark Web	[16]	No crawler used.
A Survey of the Dark Web and Dark Market Research	[101]	No specific crawler.
Exploring hackers assets: Topics of interest as indicators of compromise	[78]	No crawler used.
Dark Patterns after the GDPR: Scraping Consent Pop-ups and Demonstrating their Influence	[64]	No relevant crawler.
A credit card fake detection system using image cryptography	[81]	Irrelevant; different topic.
DSC 2018 - 2018 IEEE Conference on Dependable and Secure Computing	[62]	Only abstract; no article
12th International Conference on Security, Privacy, and Anonymity in Computation, Communication, and Storage, SpaCCS 2019	[60]	Only abstract; no article
AHFE International Conference on Human Factors in Cybersecurity, 2018	[61]	Only abstract; no article
Analyzing the DarkNetMarkets subreddit for evolutions of tools and trends using LDA topic modeling	[72]	No crawler used.
Fingerprinting web browser for tracing anonymous web attackers	[54]	No crawler used.
Halo shapes, initial shear field, and cosmic web	[82]	Irrelevant; different topic.
A web-enabled software for real-time biogas fermentation monitoring - Assessment of dark fermentations for correlations between medium conductivity and biohydrogen evolution	[43]	Irrelevant; different topic.
Onion routing circuit construction via latency graphs	[11]	No crawler used.
Financial Cryptography and Data Security - FC 2011 Workshops, RLCPS and WECSR 2011, Revised Selected Papers	[63]	Only abstract; no article

A previous research article by Bergman and Popov [7] presents a toolset called D3, developed for annotating, highlighting, collecting, and analysing.onion sites. The web crawler included in the D3 toolset was a primitive one that only served as a proof-of-concept component. To extend the D3 toolset, an adequate dark web crawler needed to be integrated into it. This chapter presents the development, integration, and testing of such a crawler.

The methodology for producing an artefact that would solve the problem of comprehensive crawling of cybercrime.onion sites was an experiments-driven design science research method (DSRM), as suggested by Perjons and Johannesson [69].

The DSRM consists of seven activities invented to guide its applier through the process of creating an artefact in a rigid yet flexible and scientific manner. The five activities are: (1) Problem Explication, (2) Requirements Definition, (3) Design and Development, (4) Artefact Demonstration, (5) Artefact Evaluation [69, p. 77].

In this research study, the problem explication was given in chapter 1. This section encloses elaborations on activities two to five. The artefact in this research was a computer program, or an *instantiation* as it is formally called, according to to Perjons and Johannesson [69, p. 29].

A. REQUIREMENTS DEFINITION

The research objectives were to integrate a crawler into the already existing toolset D3. In order to do so, a set of requirements were elicited to outline the design of the crawler. The requirements were created by the article authors themselves and complemented with previous research, as described in detail in Table 4.

B. DESIGN AND DEVELOPMENT OF ARTEFACT

It was concluded in the systematic literature review in the previous section that the most common practice amongst academics for crawling Tor onionsites was to write a program in Python with the help of the web testing library Selenium specifically for the task. Therefore the same strategy was chosen to develop the D3 complementary crawler, which resulted in a new toolset: the Digital Detectives Comprehensive Tor Toolset (DIDECT2S).

Selenium is originally a web browser automation testing and debugging tool; therefore, it typically launches a web browser to complete its tasks. The browser makes a Selenium-based crawler more powerful since it mimics human behaviour and allows human keyboard and mouse interaction. On the other hand, it also means the crawling process is slower and more computationally exhausting. Nevertheless, Selenium was chosen as the crawler's library of choice due to its flexibility and human interaction capabilities that suffice the needs of more law enforcement tasks that rarely need extensive crawls of massive amounts of .onion URLs. Since a Tor crawler requires a connection to the Tor network and to adhere to the Onion Routing protocol, it was not possible to use Selenium out of the box. Therefore, a custom-made Selenium driver, version 0.6.2, for the Tor browser [1] was used.

To enable comparison between clear web and dark web crawling and in favour of evaluating the crawler, a clear web crawler not using a Tor connection and the Tor browser was created as well. The clear web uses the same code except for the web driver and network connection. For crawling clear web pages, Selenium driver[8] version 4.4.0, using Gecko

[8] https://pypi.org/project/selenium/

TABLE 2. Summary of the 34 articles included in the systematic literature review. N/A means source code was not available.

Article	Crawler Used	Source Code	Summary
[102]	Researchers' own	N/A	A framework, "DW-GAN", for breaking CAPTCHAs, a crawler is used. Programming language not specified.
[80]	Researchers' own	N/A	A crawler used to find any differences before vs. after the Covid-19 pandemic on the dark web. Programming language: Python.
[18]	Researchers' own	N/A	Detection of crawler traps using crawling and distance measures. Programming language not specified.
[17]	SpyDark	N/A	A Tor crawler that feeds the web content to an NLP model to classify it. Programming language not specified.
[58]	Black Widow	N/A	An efficient Scrapy based Tor crawler using Apache Solr and MongoDB. Programming language: Python.
[26]	Researchers' own	N/A	A Tor crawler collecting threat intelligence data from three different dark marketplaces. Programming language not specified.
[96]	Researchers' own	N/A	Scrapy based Tor crawler for monitoring dark websites and analysing them based on a TF/IDF calculations. Programming language: Python.
[15]	TorBot	https://github.com/DedSecInside/TorBot	Comparing the "Hidden Wiki" by crawling it in 2020 and 2021. Programming language: Python.
[52]	Researchers' own	N/A	Crawled dark websites to classify web pages using ML algorithms. Programming language: Python.
[87]	CrawlBot	N/A	A researchers' own crawler for identifying child abuse material on the dark web using AI. Programming language: Python.
[3]	Researchers' own	N/A	Used a crawler to make graphs and graph calculations on Tor websites. Programming language: Python.
[6]	Researchers' own	N/A	Used a crawler to find Tor sites that were identical. Programming language: Python.
[100]	Researchers' own	N/A	Used a Selenium based crawler to classify Tor websites. Programming language: Python.
[23]	Researchers' own	N/A	A crawler was used to retrieve Tor web data and classify it after reducing its feature space. Programming language: Python.
[50]	Researchers' own	N/A	A Selenium based crawler was used to harvest data specific to South Korean dark websites. Programming language not specified.
[99]	Unclear	N/A	A Hadoop based framework for collecting and analysing Tor web content. Programming language not specified.
[86]	Researchers' own	N/A	A Selenium based crawler to identify IoT attack trends. Programming language: Python.
[47]	ACHE	https://github.com/ViDA-NYU/ache	Clear and dark web crawler used to identify IoT threat intelligence. Programming language: Java.
[55]	(1) Researchers' own (2) Branwen et al. [10]	N/A	Crawled and analysed data to investigate relation between drug overdoses and drug ads using Scrapy. Programming language: Python.
[68]	Researchers' own	N/A	A Selenium based crawler used to identify characteristics of Tor websites over time. Programming language: Python.
[85]	The Dark Crawler	N/A	A crawler used for sentiment analysis of data for identifying extremist content. Programming language not specified.
[95]	Researchers' own	Upon request	Crawling and scraping dark marketplaces with researchers' own Selenium based crawlers. Programming language: Python.
[67]	Researchers' own	N/A	A conceptual system for crawling suspicious and malicious .onion sites. Programming language not specified.
[31]	Researchers' own	N/A	A crawler developed for investigating dark marketplaces. Programming language: AppleScript.
[40]	Researchers' own (based on Nutch)	N/A	A clear and dark web crawler framework for classifying content using ML algorithms. Java.
[77]	(1) [34] (2) [57]	N/A	A crawler used to classify content based on fuzzy kNN. Programming language: PHP and Python.
[12]	(1) bUbiNG (2) Researchers' own	https://github.com/LAW-Unimi/BUbiNG	A suite for Tor crawling and text mining, customised using Scrapy spiders. Programming language: Python.
[88]	Researchers' own	N/A	By using a Selenium based crawler through Tor, discrimination of Tor exit nodes is examined. Programming language not specified.
[27]	OnionCrawler	N/A	Automatic thematic labelling of Tor web content crawled using researchers' own OnionCrawler. Programming language not specified.
[44]	Researchers' own	N/A	Analysis of illicit products using AI algorithms based on crawled Tor web content using Nightmare.js. Programing language: JavaScript
[103]	Dark Crawler	N/A	Analysis of extremist content fetched using a Tor crawler based on [9]. Programming language not specified.
[2]	DATACRYPTO	N/A	Analysis of monthly revenue per drug on Silk Road 1.0. Used Researchers' own crawler. Programming language not specified.
[42]	Nutch	https://github.com/apache/nutch	A customised Nutch crawler automatically classifying explosives content from Tor. Programming language: Java.
[25]	Researchers' own	N/A	Crawling Tor web to identify extremist content. Programming language not specified.

TABLE 3. Articles excluded in the data extraction activity. Reason for exclusion in the right-hand column.

Title	Reference	Reason for Exclusion
Big Data, Method and the Ethics of Location: A Case Study of a Hookup App for Men Who Have Sex with Men	[53]	Not concerning any ACN such as Tor, I2P, or Freenet.
Inference in OSNs via lightweight partial crawls	[4]	No crawler.
Discovering Topics from Dark Websites	[98]	Not concerning any ACN such as Tor, I2P, or Freenet.
Dark Web—Onion Hidden Service Discovery and Crawling for Profiling Morphing, Unstructured Crime and Vulnerabilities Prediction	[79]	Theoretical work; no crawler used.
The investigation of the possibility of automated collection of information in the hidden segment of the Internet	[51]	Not concerning any ACN such as Tor, I2P, or Freenet.
Discovering abnormal behaviors via HTTP header fields measurement	[28]	Not concerning any ACN such as Tor, I2P, or Freenet.
Understanding website behavior based on user agent	[70]	Tor network and crawlers addressed separately.

TABLE 4. Requirements elicited by the author for the building a dark web crawler to be integrated with the already existing toolset.

ID	Requirement	Comments
RQ1	Support for crawling .onion services.	Reference: Authors themselves, [71]
RQ2	Option to scrape JavaScript	Most onion sites do not allow JS [18], although it should be possible.
RQ3	Support for graphic content retrieval	Reference: Graphic content is crucial in CSAM investigations [22]
RQ4	Support credentials authentication (e.g. cookies, username/password).	Reference: [71]
RQ5	Support for solving CAPTCHA token and authentication gates.	Reference: [71]
RQ6	Support for automatic screenshots for completeness and forensic soundness.	Reference: Authors themselves.
RQ7	Support for parallel processing.	To reduce download time. Reference: [71]
RQ8	Support extensive logging to maintain the chain of custody.	Reference: Authors themselves.
RQ9	Support for local or in-house hosting; non-cloud based.	Reference: Authors themselves.
RQ10	Be of open source code.	Reference: Authors themselves.
RQ11	Support delay in crawling	To avoid crawler traps [18].

driver version 0.31.0 for Linux 64-bit,[9] and a non-modified Internet connection from Stockholm University was used.

In simple terms, the logic of the crawler was the following:

1) Initiate log file
2) Go to URL home page
3) Get robots.txt (if relevant)
4) Take screenshot
5) Save home page source
6) Find link elements on home page
7) For each link that is not external domain or disallowed by robots.txt
   a) Request link web page
   b) Save page source
   c) Save images on page
   d) Take screenshot
8) Close all file handlers and log files and quit program

It should be noted that this crawler was designed for a laboratory environment. Thus it was configured to obey robots.txt and not crawl external URLs to avoid fetching unwanted content. Furthermore, the crawler was designed to be general rather than a crawler tailored for a specific website. Therefore it might not work as intended on all websites, depending on how the website is constructed.

C. LIMITATIONS OF THE ARTEFACT

The developed crawler was characterised by a few limitations. Firstly, it was built for cybercrime investigation and digital forensic purposes with forensic soundness and correctness primarily in mind. Secondly, the performance speed was not a priority. The Selenium-based web driver used for the developed crawlers limits the execution speed since it operates an actual browser that requires multiple system libraries and components to launch to operate. In addition, crawling the Tor network is intrinsically slower than the regular Internet due to the onion routing, which limits the execution speed of Tor crawlers in general.

D. DEMONSTRATION OF THE ARTEFACT

The artefact is demonstrated in this activity of the design science research method. In this case, a web crawler was built to complement and extend an existing toolset. Figure 2 presents the updated toolset called DIDECT2S. The crawler is highlighted in red.

The logic of the program was briefly explained in the previous section; however, the complete source code is available on: https://gitea.dsv.su.se/jebe8883/DIDECT2S.

E. EVALUATION OF THE ARTEFACT

The artefact designed and developed based on the outcomes of an exhaustive systematic literature review was a comprehensive Tor web crawler integrated into an already existing toolset of dark web cybercrime investigative tools.

[9] https://github.com/mozilla/geckodriver/releases/download/v0.31.0/geckodriver-v0.31.0-linux64.tar.gz

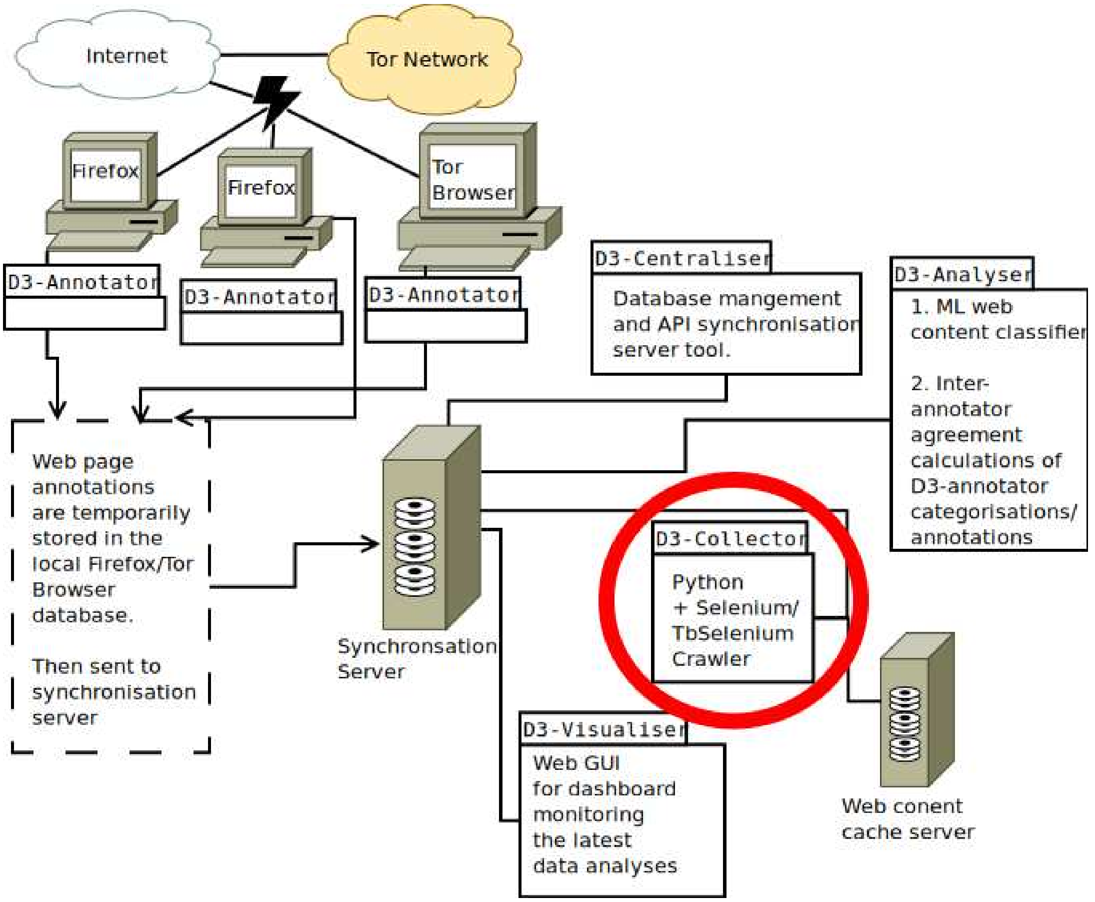

Internet
Tor Network
Firefox
Firefox
Tor Browser
D3-Annotator
D3-Annotator
D3-Annotator
D3-Centraliser
Database mangement and API synchronisation server tool.
D3-Analyser
1. ML web content classifier
2. Inter-annotator agreement calculations of D3-annotator categorisations/ annotations
Web page annotations are temporarily stored in the local Firefox/Tor Browser database.
Then sent to synchronisation server
Synchronsation Server
D3-Collector
Python + Selenium/ TbSelenium Crawler
D3-Visualiser
Web GUI for dashboard monitoring the latest data analyses
Web conent cache server

FIGURE 2. Topology of the updated D3 toolset - the Digital Detective's Comprehensive Tor Toolset (DIDEC2TS) with the new crawler component highlighted in red.

To concretely exhibit how well the developed artefact fulfilled the requirements specified for it, this section presents a requirement and artefact evaluation as the final activity in the design science research method.

A couple of experiments were conducted to evaluate the artefact, confirm its requirement fulfilment, and assess its overall usefulness and effectiveness. The experiments were done in a laboratory environment setting using realistic case scenarios and authentic websites as experiment objects.

The experiment was divided into two parts: the first was to crawl both clear and dark websites, and the second was to crawl only a dark marketplace protected by authentication and CAPTCHA on the Tor network. Both experiments were designed to verify that the underlying requirements of the artefact were met and fulfilled.

Non-functional requirements such as RQ11 - that the software should be open source, RQ1 - that the crawler should support crawling .onion addresses, and RQ9 - that it should be possible to self-host the crawler were implicitly fulfilled as the crawler was built and the source code was published. Similarly, RQ7 - support for parallel processing was considered fulfilled since the Selenium web driver, which was used to build the crawler, can be executed in parallel according to its official documentation [74]. All other functional requirements were affirmed in the experiment, as depicted in Figure 3.

The first part of the experiment consisted of crawling the same set of websites located on both the regular Internet (clear web) and the Tor network (dark web) to verify that the Tor connection and crawling mechanisms work just as well as the clear web ditto. The web pages crawled on the regular Internet should ideally be the same as those crawled on the Tor network, given that the crawler works as intended. Of course, some content might be updated or changed between the crawls, but to a large extent, the saved web pages should be the same since the website is the same.

Websites are hosted on web servers on both the regular Internet, as well as on the Tor network. On Tor, web servers are, as mentioned, called Onion Services, and the websites are referred to as onionsites. The terms used in this article will be "clear websites" and "clear web", and "dark websites" and "dark web" respectively. The websites used in the evaluation experiment of the crawler were the following:

1) **Debian**
   - https://debian.org/
   - http://5ekxbftvqg26oir5wle3p27ax3wksbxcecnm6oemju7bjra2pn26s3qd.onion

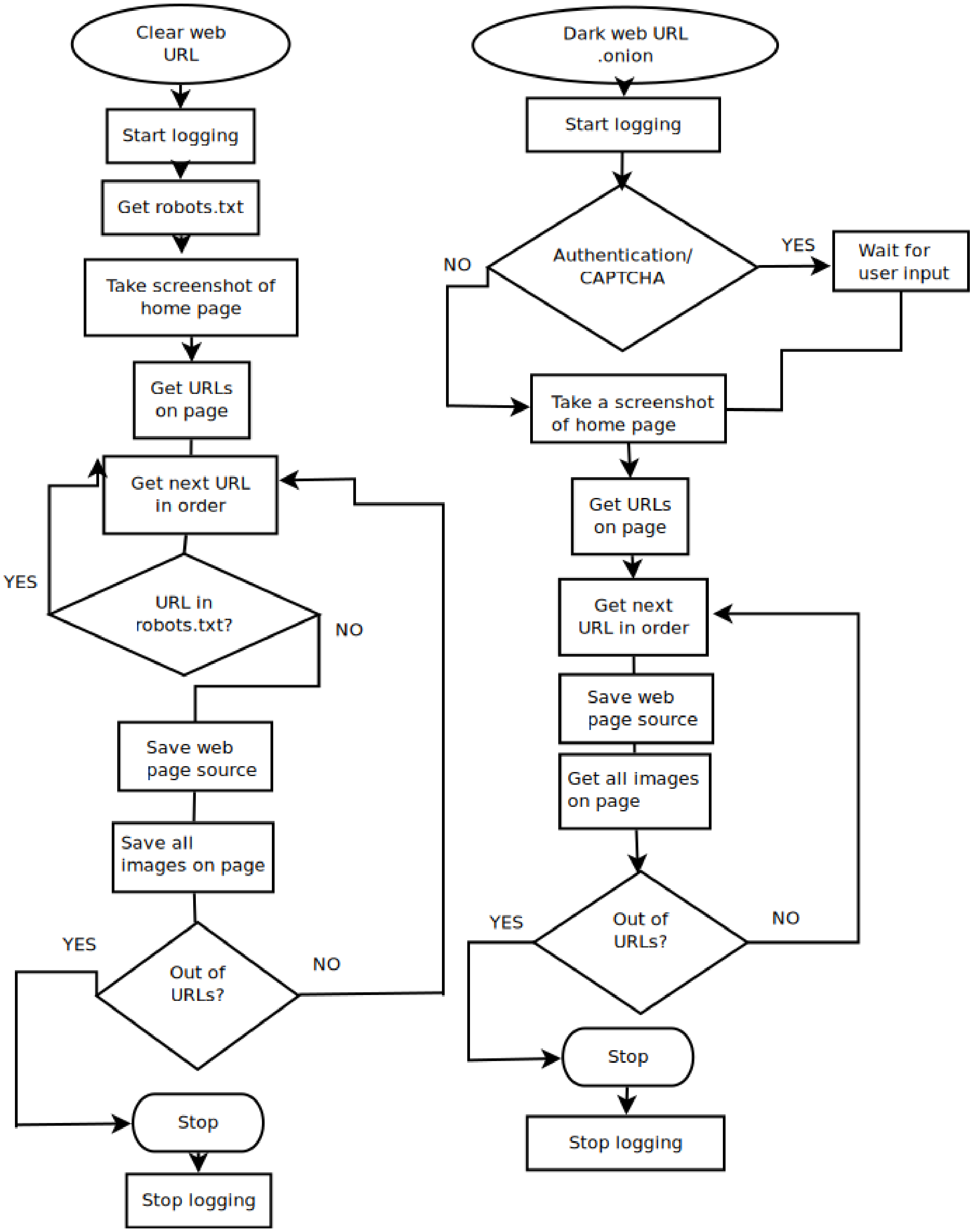


FIGURE 3. Flowchart depicting the two experiment scenarios in the artefact evaluation. The clear website crawling is presented on the left hand side and the dark website crawling on the right hand side.

2) **The Guardian**
   - https://theguardian.com/
   - https://www.guardian2zotagl6tmjucg3lrhxdk4dw3lhbqnkvvkywawy3oqfoprid.onion
3) **New York Times**
   - https://www.nytimes.com/
   - https://www.nytimesn7cgmftshazwhfgzm37qxb44r64ytbb2dj3 × 62d2lljsciiyd.onion
4) **Qube-OS**
   - https://www.qubes-os.org/
   - http://qubesosfasa4zl44o4tws22di6kepyzfeqv3tg4e3ztknltfxqrymdad.onion
5) **CIA**
   - https://cia.gov/
   - http://ciadotgov4sjwlzihbbgxnqg3xiyrg7so2r2o3lt5wz5ypk4sxyjstad.onion

The second part of the experiment was to assess the crawler's capabilities further. The crawler was set to crawl a dark marketplace protected by username and password authentication as well as a CAPTCHA token. This part of the experiment served the purpose of verifying that the crawler could solve a typical dark web investigation task of saving clues and evidence from, for example, a dark marketplace or a child abuse website behind an authentication portal.

Since there was no existing comparison data set for the scraped web content in the experiment, manual verification of the data downloaded was done. The dark marketplace that was chosen for this crawling task was White House Market (WHM) http://hvilngbbx2yxtq7ilsrjsosv374phq4jx2nq5izot 5baxlqyjy3u2cid.onion. The White House Market has been available on the Tor network since 2019 and has over 3000 vendors, according to the website.

WHM was considered a typical target for cybercrime investigation since it required authentication and CAPTCHA solving, and the content of interest hosted on it was both text and images. In short, it was deemed a representative website to scrape for examining the fitness and performance of the developed crawler. An account was created to log in to and scrape the WHM homepage.

As a first step, a manual inspection of the website was done to assess its structure and content. It was concluded that there was no sensitive image material or private information such as email addresses or personal IDs that, according to the ethical research codex, should not be downloaded, analysed, or processed without consideration. The WHM pages subject to scraping merely contained usernames, user avatars, product descriptions, prices, and other (primarily illegal) product META data.

The crawler built was instructed for all tasks to fetch all pages from the starting page, but not more than that. This limitation was set not to overload the web server or avoid being blocked from the website for the purpose of the experiment. In addition, a delay between each request was configured not to overload the servers more than necessary and avoid being blocked by the web server. Python's random library[10] was used to pseudo-randomly generate a delay of zero to four seconds between each request. Both the clear web and the dark web crawler were configured to, by default, collect and download all links and images on the web page it was given when it started.

1) EVALUATION OF CRAWLING CLEAR- AND DARK WEBSITE PAIRS

To make the clear web and dark web crawls as similar as possible in the first part of the experiment, the browsers were configured with the same settings and add-ons to avoid any discrepancies in the results due to misconfiguration. In an authentic cybercrime investigation, the investigators would not respect the robots.txt file when crawling; however, in this experiment, the robots.txt were respected, and none of the disallowed entries were scraped by the crawler for ethical reasons.

To evaluate how well the crawler fetched web pages from the same websites on both the clear and the dark web, each pair of pages were compared by measuring their degree of similarity. Each pair consisted of a web page from the crawled clear website, and the same web page crawled from the dark website, e.g. https://guardian.com/index.html and https://guardian.onion/index.html comprise a pair in this step of the evaluation.

The comparison of retrieved web pages was both manual and computational. Firstly, the directory contents were manually inspected to determine which files were downloaded from each crawler. Secondly, the file content differences were manually inspected by the researchers using GNU command line program Diffutils.[11] To count the number of files that differ, GNU Wc word counter[12] was used. For explicitly extracting the files that differed and not including the ones that were the same, GNU Grep[13] was used in combination with Diff and Wc. The grep keyword was changed to "same" to find the files that Diff reported as identical. This resulted in a binary classification of files that differed and files that did not. In the next step, the similarity score would reveal to what degree the files differ. The full command reads:

```
$ diff -q -s -N website-CW/webpage.html
website-DW/webpage.html |  grep ``diff''
| wc -l
```

The computational comparison was one using a Python script that converted the websites' textual content to a vector of characters and then calculated the similarity measures between the vectors. In information retrieval, different similarity measuring algorithms are used to compare a search query string with a retrieved document. The most common algorithms used include: Cosine, Euclidean, Jaccard, and Okapi [29].

Similarity measure algorithms are often used on written "human language" where semantics are crucial. However, in this experiment, the text documents to be compared consisted of "machine language" - HTML, JavaScript, and CSS, therefore they documents were converted into character based vectors instead of word-based vectors, which would not take into account machine language characters.

The cosine similarity score is calculated from the cosine angle between two documents represented as vectors, e.g. A and B, with their inner product divided by the vector product of A and B. Formally expressed as:

$$\cos\theta = \frac{A \cdot B}{|\vec{A}||\vec{B}|}$$

[10] https://docs.python.org/3/library/random.html

[11] https://www.gnu.org/software/diffutils/

[12] https://www.gnu.org/software/coreutils/manual/html_node/wc-invocation.html

[13] https://www.gnu.org/software/grep/manual/html_node/index.html

Each web page was converted into a vector of characters and frequency using Scikit's CountVectorizer.[14] As a second step in the process, the cosine similarity[15] and Jaccard similarity scores[16] were calculated for each clear web page and the dark web ditto. The clear web and dark web pages with the same title and filename were considered a "pair". Orphan, i.e. single web pages with no clear- or dark web "partner", were excluded from the similarity calculation. An example of a clear web-dark web page pair is the support.html page of Debian's websites (curly brackets are only a pair indicator):

```
{https://debian.org/support.html,
http://5ekxbftvqg26oir5wle3p27ax3wksbxce
cnm6oemju7bjra2pn26s3qd.onion/support.
html}
```

As a final step in the clear and dark website comparison, the images saved by the crawler from the website pairs were compared in "image pairs", similar to the web page pairs. The purpose of this was to confirm whether they were exactly the same or not. Textual web page content, such as HTML and JavaScript, is more changeable than pictures since it might change for different locations, browser agents, IP addresses, or the current time; images are usually less dynamic. For this reason, a hash sum comparison between images retrieved from the clear- and dark websites was deemed adequate to estimate any differences in content retrieval discrepancy.

The comparison of clear web and dark web images was made by calculating the SHA1 hash sum for each respective image in the pair. Practically this was done with the with Sha1deep[17] as follows:

```
$ sha1deep -m website-CW/*.jpg >
cw_hash_sums.txt
$ sha1deep -m cw_hash_sums.txt
website_DW/*.jpg
```

2) EVALUATION OF CRAWLING A DARK MARKETPLACE

The crawler was evaluated on a dark marketplace crawling task as a second part of the artefact evaluation. Since there was ground truth data set to compare the crawled dark marketplace data, it was manually verified that the crawler had scraped all available links. There were too many images to verify that they were correctly fetched by the crawler manually; therefore only images from the starting page were verified.

The crawler was set to crawl the White House Market, scrape all links, and download all link pages and images on the starting page. Since WHM does not, for security reasons, allow JavaScript to be enabled, it was disabled in the Selenium Tor Browser. After the scraping was done, the White House Market's homepage source was manually inspected to verify that all links were fetched by the crawler.

V. RESULTS

The results from the first part of this research study, the systematic literature review, were presented in the previous section. In this section, the results from the implementation and evaluation of the developed clear web and dark web crawler is presented.

A. CLEAR WEB AND DARK WEB CRAWLING RESULTS

The crawler was implemented for scraping both clear web and dark websites, and the data collected from each web type was compared using a couple of different techniques and measures.

First, the semi-manual inspection of the website pairs was done using GNU Diffutils. Diffutils identified discrepancies between the scraped web content files. Table 5 shows the number of pages that were downloaded from the clear web (CW) and the dark web (DW) versions of the websites in question. The web pages were saved as local files by the crawler, and the third column presents how many of the files had identical content. The fourth column presents how many of the file names - in this study, equivalent to the titles of the web pages when downloaded - were identical. The duration of the crawling process is found in column five in Table 5.

The results show that the crawler fetched the same number of pages from both the clear web- and the dark websites for Debian, QubeOS, and CIA. In the case of CIA's website, however, the index.html was downloaded twice from the clear web crawler. This issue was due to a programmatic error related to the website's usage of internal URLs.

Concerning The Guardian's websites, there were 12 files that were not retrieved from The Guardian's onionsite. The files missing from scraping The Guardian's onionsite were URLs available only over their clear website; see Figure 4. The random wait for The Guardian was set to 0-4 seconds to avoid blocking, and therefore the complete scraping of the 201 web pages took circa 26 minutes. The scraping of the clear web version of The Guardian took circa six minutes with the same random delay of 0-4 seconds between each HTTP request.

The New York Times' onionsite blocked the dark web crawler from collecting certain pages, as seen in Figure 5. Due to this data collection disruption, there were pages collected from the clear website but not the dark website. According to the message displayed, the crawler was blocked based on its IP address, and the server required a CAPTCHA to be solved in order to continue. The reason for blocking the crawler was most likely that the IP address was shared with other Tor users, as opposed to the clear web crawler, which used Stockholm University's IP address range.

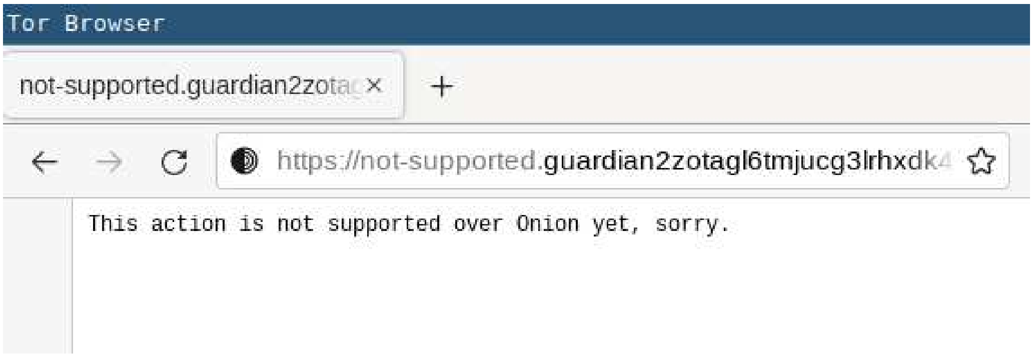
The similarity scores were calculated for each clear web-dark web page pair. Due to the vast number of web page pairs, only the highest, lowest, mean, and median of the similarity scores for each website are presented in Table 6. Note that

[14] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

[15] https://numpy.org/doc/stable/reference/generated/numpy.cos.html

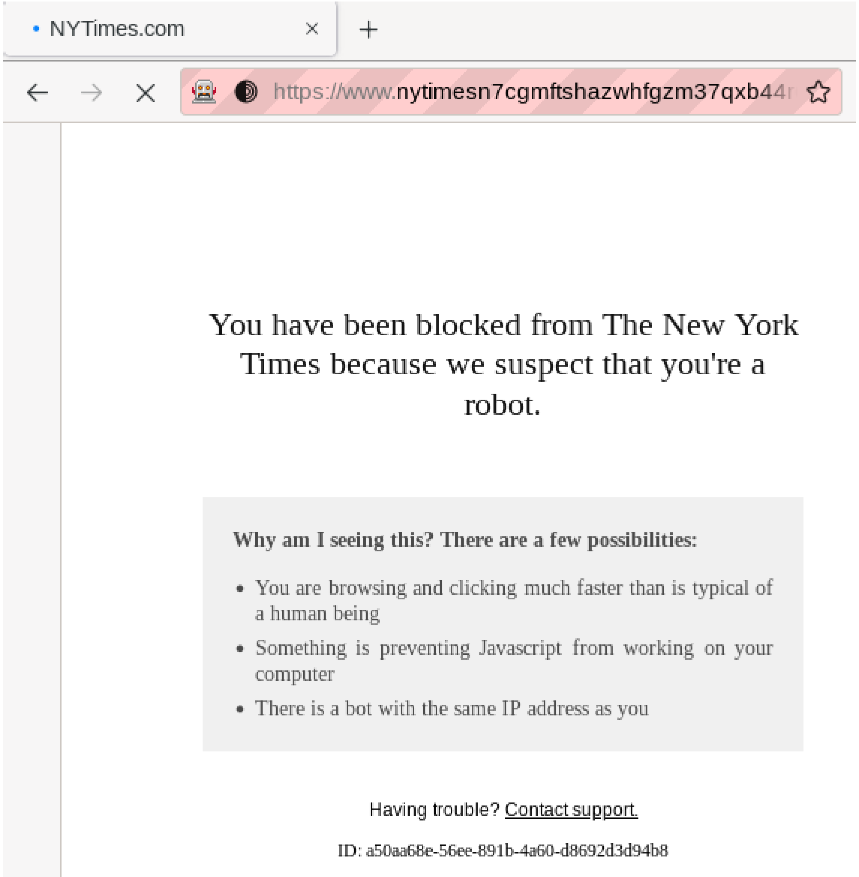
[16] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.jaccard_score.html

[17] http://md5deep.sourceforge.net/


Tor Browser
not-supported.guardian2zota
https://not-supported.guardian2zotagl6tmjucg3lrhxdk

This action is not supported over Onion yet, sorry.

FIGURE 4. Screenshot of a web page that was unavailable on the The Guardian's Tor website.



You have been blocked from The New York Times because we suspect that you're a robot.

Why am I seeing this? There are a few possibilities:

- You are browsing and clicking much faster than is typical of a human being
- Something is preventing Javascript from working on your computer
- There is a bot with the same IP address as you

Having trouble? Contact support.

ID: a50aa68e-56ee-891b-4a60-d8692d3d94b8

FIGURE 5. Screenshot of a web page shown when the crawler was blocked from The New York Times Tor website based on its Tor IP address.

TABLE 5. Number of pages downloaded from each website, the number of identical files, the number of files with the same title and file name, and the timestamps of the execution of the crawls.

CW/DW Website	Web Pages (CW, DW)	Identical Files	Identical File Names	Date, Timestamp(CW,DW) (CET)
Debian	50, 50	45	49	2022-09-09, 09:26:02-09:26:40, 09:09:13-09:21:31
Qube-OS	64, 64	17	61	2022-09-01, 13:15:57-13:18:40, 13:46:10-13:58:57
The Guardian	223, 211	0	204	2022-09-09, 10:16:43-10:24:13, 09:40:10-10:13:29
New York Times	200, 78	0	49	2022-09-16, 11:50:32-12:04:55, 11:18:14-11:48:12
CIA	42, 41	1	32	2022-09-09, 11:06:44-11:08:41, 11:18:57-12:14:46

the cosine similarity score is abbreviated as "CS". The exact scores for each pair can be found online.[18] As can be seen, the cosine similarity mean for all web page comparisons ranges between 0.9698 and 0.9999.

In addition to web pages, images were downloaded by the crawler. Since there was a discrepancy between the number

[18] https://gitea.dsv.su.se/jebe8883/SLR/

TABLE 6. Cosine (CS) similarity scores for the least similar and most similar web pages scraped from each web type of the same website - the clear (CW) and dark (DW) websites respectively.

CW/DW Website	Lowest (CS)	Highest (CS)	Mean (CS)	Median (CS)
Debian	0.9977	1.0	0.9999	0.9999
Qube-OS	0.9926	1.0	0.9966	0.9999
The Guardian	0.5584	0.9999	0.9927	0.9985
New York Times	0.3056	0.9999	0.9698	0.9981
CIA	0.8824	1.0	0.9842	0.9998

of downloaded web pages from the clear web and dark web, respectively, there was naturally a discrepancy between the number of images downloaded from each website. The number of images retrieved from each respective website is presented in Table 7.

B. DARK MARKETPLACE CRAWLING RESULTS

The DIDECT2S dark web crawler was used to crawl a dark marketplace in order to demonstrate and validate that fits its purpose as a digital investigation tool.

The crawler was tasked to scrape all images, links, and linked pages from the home page of the dark marketplace White House Market onionsite. At the time of scraping, September 2022, the manual observation found 251 links on the homepage of the dark marketplace in question. The crawler managed to scrape the pages for all links identified on the home page and the images on those pages. Details can be found in table 8.

In total, 250 pages, including 2881 images, were fetched in 20 minutes and 47 seconds, including a request delay of zero to four seconds. On average, the crawler scraped 12 web pages per minute, including source code and images. The image sizes varied between 5758 bytes and 663 bytes.

VI. DISCUSSION

This research article was divided into two segments: one theoretical systematic literature review and one practical design science implementation based on the theoretical findings. The results from the systematic literature review in segment one concluded that the programming language Python, in combination with the web debugging and scraping library Selenium was the most used combination for developing dark web crawlers in academic studies. Consequently, the experiment results from the developed Tor web crawler in segment two demonstrated that it was a compelling duo.

The systematic literature review results showed that researchers built their own crawlers in most scientific articles concerning dark web crawling. Moreover, the results showed that few crawlers were released publicly as open-source code; only four out of 34 did so. The absence of open-source crawlers conforms with the results from previous research by Kumar, Bhatia, and Rattan [49], namely that few researchers use open-source crawlers and that few researchers mention which open-source crawler was used in their study.

The practical implementation of a crawler, based on the theoretical findings, was evaluated in an experiment. Experiments cannot be generalised and are not applicable to the real, constantly changing world. However, the results indicate that the crawler developed and presented is usable, effective, and, under certain circumstances, efficient.

The experiments were successfully completed, and the functional requirements were consequently assessed and fulfilled. The results from the experiments also showed that the crawler was capable of crawling both clear web and dark websites. Due to the fact that two different Selenium drivers were used for each type of web, the experiment evaluation verified that both worked correctly, i.e., web pages and images were fetched in their entirety from the websites requested.

When the implemented crawler scraped websites available on both the Tor network and the regular Internet, many of the same pages and the same number of pages were fetched. In the case of Debian's dark and clear websites, 45 out of 50 the web pages were identical; 61 out of 64 from Qubes-OS websites, and 204 of the web pages fetched from the Guardian's websites had the same URLs, except for the domain (.com and .onion respectively), although the content partially differed. The numbers of pages fetched from the New York Times' websites were very far apart since the crawler was blocked from the New York Times' onionsite. As a whole, the crawler managed to scrape the onionsites just as well as the clear websites; indicating that the Tor connection and crawler logic worked as intended.

The cosine similarity scores indicated that the content for some pages differed. However, there was a high similarity for the web pages that comprised pairs; at the lowest, the mean cosine similarity was 0.9698. The main point with the similarity scores was to verify that the dark web and clear web pages were essentially the same and that the crawlers achieved similar results.

The images downloaded by the crawler differed notably between the clear and dark websites of the Guardian and the New York Times websites. The log file of the crawler indicated that few of the image elements found were actually downloaded, most likely due to the fact that the image resources the Guardian and New York Times onionsites were located on the clear web, where Tor exit node IP addresses were blocked from retrieving content. However, this does not affect onionsite crawling in the cybercrime context too much

TABLE 7. The total number of images downloaded by the crawler from each respective website together with the number of images that differed between the websites in each website pair.

CW/DW Website	Number of Images (CW, DW)	Identical Images	Difference
Debian	15, 15	14	1 different picture
Qube-OS	56, 56	38	18 different pictures
The Guardian	95 , 52	12	135 different pictures
The New York Times	77, 21	0	98 different pictures
CIA	24, 24	10	14 different pictures

TABLE 8. Details of a scraping of a dark marketplace website using the implemented crawler.

Timestamps	2022-09-08, 13:58:58-14:18:45 (CET)
Duration	20 minutes 13 seconds
Pseudo-random page request delay interval	0-4 seconds
Links on homepage	251
Downloaded web pages	251
Unique images downloaded	2881
URL	http://hvilngbbx2yxtq7ilsrjsosv374phq4jx2nq5izot5baxlqyjy3u2cid.onion

since illicit and illegal onionsites seldom redirect to external resources as news websites do.

The crawling library Selenium was originally built as a web testing tool but has rendered into an effective web scraping tool. However, it is not the most efficient library for web crawling and web scraping in terms of speed, although it has a few advantages that are relevant to digital investigations: (1) it allows user interaction, (2) it mimics human behaviour well, (3) it works visually and hence gives an investigator the option to watch as the crawler fetches each page; in this way, the process is ''invigilated''. Arguably, this would increase the credibility of the scraping process as a means of collecting potential clues and evidence, subject to court admissibility.

Furthermore, by using Selenium, the crawler operator has the option to manually tweak and configure settings during runtime, such as bookmarking a page, clicking buttons, go back or refresh a page, enable or disable JavaScript, or establish a new Tor circuit.

The performance in regards to speed was for the Tor crawler slower than the clear web crawler, much because the Tor network is slower than the regular Internet that does not use the Onion Routing protocol. In addition, a delay of one to four seconds for fetching web pages was programmed into the crawlers in order to avoid being blocked by the web servers, although this was not an entirely successful action since The New York Times' onionsite blocked the crawler, as previously mentioned.

The results from the experiments showed that the Tor crawler managed to scrape 251 pages in 20 minutes, i.e. 12 pages per minute from the dark marketplace White House Market. In contrast, it took around eight minutes to scrape The Guardian's clear website of 223 pages, equal to 27.8 web pages per minute. Scraping 211 pages from The Guardian's onion took circa 33 minutes - an average of circa 6.4 pages per minute. In summary, the dark web crawler was more than four times slower.

Performance figures for crawlers like the one designed and implemented in this research article include metrics such as the number of pages retrieved and accuracy rates of web content classification. As opposed to clear web crawling, time alone is not a relevant performance metric for dark web crawling due to its resource-intense encryption and routing scheme that requires more time and power than clear web page fetching. Only one out of 44 articles mention crawling duration and retrieval of web pages per minute and how it increases with additional instances of browsers running the crawler, namely [68].

However, the clear web crawler built in this research could still be compared to other clear web crawlers. According to Kumar, Bhatia, and Rattan [49], Mercator is one of the fastest open-source crawlers, capable of retrieving 112 pages per second in 1999 [32]. Numerous modern crawlers reportedly are faster than the crawler presented in this paper, which averages 0.46 web pages per second. However, it should be noted that the hardware requirements differ, as well as the software libraries used. The library used in this research was Selenium, which runs single-threaded in a graphical interface browser for usability reasons; this is a significantly slower architecture than multi-threaded distributed crawling engines used by, for example, the Internet Archive or big search engine companies. In this case, there is a trade-off between speed and usability, where the crawler presented in this research favoured usability over speed to fit its ultimate purpose.

Nevertheless, performance is not measured only in execution speed but also in efficacy and accuracy. Comparing efficacy between different research works and crawlers under different circumstances, using different is not optimal. Thus, benchmarking it against other research figures was deemed inadequate for this study. However, the execution times combined with the cosine similarity scores for the pages downloaded by the crawler will hopefully provide an indication of its effectiveness.

VII. CONCLUSION AND FUTURE RESEARCH

The current research article addresses the problem with data collection in cybercrime cases in general and dark web-related cybercrime cases in particular. The research problem presented pointed out the need for data collection tools in dark web investigations and suggested a solution to the problem by presenting a prototype that fulfilled a number of requirements for such a dark web investigative software tool. The scientific foundation that preceded the development of the proposed software consisted of a systematic literature review that included 58 research articles concerning crawling the dark web.

The primary purpose of this research study was to establish knowledge regarding dark web crawlers in academic research. From this knowledge, a dark web crawler was developed to fit into a pre-existing dark web cybercrime toolset called D3.

In combination with an annotation-based machine learning classifier in the D3 toolset, the crawler developed and presented in this article will capacitate the toolset to automatically collect and classify web content based on previously annotated web pages. Ultimately, this will save manual labour for cybercrime investigators going through large quantities of web content while not losing control over the crawling process. Neither will it compromise the forensic soundness of the overall process since a certain amount of operator presence and interaction is necessary to use it. For example, URL selection, crawling configuration, and user authentication might require user interaction. A logical continuation of this research would be to further elaborate on and test the toolset and make an expert or user evaluation of it.

In addition, a further assessment of crawler blocking mechanisms would be essential to establish a methodology for improving crawler performance on Tor network, despite its natural limitations. It can be assumed that some onionsite administrators and programmers will improve their crawler-blocking mechanisms and strengthen their authentication mechanisms in the future. Given that there is no competence deficiency amongst unethical and criminal web developers, the digital cat-and-mouse game will persist.

ACKNOWLEDGMENT

The literature review data was downloaded from Scopus API via http://api.elsevier.com and http://www.scopus.com.

REFERENCES

[1] G. Acar and M. Juarez. (2020). *Individual Contributors. TOR-Browser-Selenium TOR Browser Automation With Selenium*. [Online]. Available: https://github.com/webfp/tor-browser-selenium

[2] J. Aldridge and D. Décary-Hétu, "Hidden wholesale: The drug diffusing capacity of online drug cryptomarkets," *Int. J. Drug Policy*, vol. 35, pp. 7–15, Sep. 2016, doi: 10.1016/j.drugpo.2016.04.020.

[3] A. Alharbi, M. Faizan, W. Alosaimi, H. Alyami, A. Agrawal, R. Kumar, and R. A. Khan, "Exploring the topological properties of the Tor dark web," *IEEE Access*, vol. 9, pp. 21746–21758, 2021, doi: 10.1109/access.2021.3055532.

[4] K. Avrachenkov, B. Ribeiro, and J. K. Sreedharan, "Inference in OSNs via lightweight partial crawls," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 44, no. 1, pp. 165–177, Jun. 2016, doi: 10.1145/2964791.2901477.

[5] A. Baravalle, M. S. Lopez, and S. W. Lee, "Mining the dark web: Drugs and fake ids," in *Proc. IEEE 16th Int. Conf. Data Mining Workshops (ICDMW)*, Dec. 2016, pp. 350–356, doi: 10.1109/ICDMW.2016.0056.

[6] F. Barr-Smith and J. Wright, "Phishing with a darknet: Imitation of onion services," in *Proc. APWG Symp. Electron. Crime Res. (eCrime)*, Nov. 2020, pp. 1–13, doi: 10.1109/ecrime51433.2020.9493262.

[7] J. Bergman and O. B. Popov, "The digital detective's discourse—A toolset for forensically sound collaborative dark web content annotation and collection," *J. Digit. Forensics, Secur. Law*, vol. 17, no. 5, pp. 1–25, doi: 10.15394/jdfsl.2022.1740.

[8] M. Bernaschi, A. Celestini, S. Guarino, F. Lombardi, and E. Mastrostefano, "Spiders like onions: On the network of Tor hidden services," in *Proc. World Wide Web Conf.*, May 2019, pp. 105–115, doi: 10.1145/3308558.3313687.

[9] M. Bouchard, K. Joffres, and R. Frank, "Preliminary analytical considerations in designing a terrorism and extremism online network extractor," in *Computational Models of Complex Systems*. Cham, Switzerland: Springer, 2014, pp. 171–184, doi: 10.1007/978-3-319-01285-811.

[10] G. Branwen. (Jul. 2015). *Dark Net Market Archives*. Accessed: Feb. 2, 2020. [Online]. Available: https://www.gwern.net/DNM-archives.dataset

[11] S. Castillo-Pérez and J. Garcia-Alfaro, "Onion routing circuit construction via latency graphs," *Comput. & Secur.*, vol. 37, pp. 197–214, Sep. 2013, doi: 10.1016/j.cose.2013.03.003.

[12] A. Celestini and S. Guarino, "Design, implementation and test of a flexible Tor-oriented web mining toolkit," in *Proc. 7th Int. Conf. Web Intell., Mining Semantics*, Jun. 2017, pp. 1–10, doi: 10.1145/3102254.3102266.

[13] E. Crowder and J. Lansiquot, "Darknet data mining—A Canadian cybercrime perspective," 2021, *arXiv:2105.13957*.

[14] V. D. D'Agostino. (2014). *Complaint: United States of America V. Blake Benthall*. Accessed: May 5, 2022. [Online]. Available: https://www.justice.gov/usao/nys/pressreleases/November14/BlakeBenthallArrestPR/Benthall%2C%20Blake%20Complaint.pdf

[15] A. Dalvi, L. Ankamwar, O. Sargar, F. Kazi, and S. G. Bhirud, "From hidden Wiki 2020 to hidden Wiki 2021: What dark web researchers comprehend with Tor directory services?" in *Proc. 5th Int. Conf. Inf. Syst. Comput. Netw. (ISCON)*, Oct. 2021, pp. 1–4, doi: 10.1109/iscon52037.2021.9702384.

[16] A. Dalvi, I. Siddavatam, V. Thakkar, A. Jain, F. Kazi, and S. Bhirud, "Link harvesting on the dark web," in *Proc. IEEE Bombay Sect. Signature Conf. (IBSSC)*, Nov. 2021, pp. 1–5, doi: 10.1109/ibssc53889.2021.9673428.

[17] A. Dalvi, "SpyDark: Surface and dark web crawler," in *Proc. 2nd Int. Conf. Secure Cyber Comput. Commun. (ICSCCC)*, May 2021, pp. 45–49, doi: 10.1109/icsccc51823.2021.9478098.

[18] B. David, M. Delong, and E. Filiol, "Detection of crawler traps: Formalization and implementation—Defeating protection on internet and on the TOR network," *J. Comput. Virol. Hacking Techn.*, vol. 17, no. 3, pp. 185–198, Sep. 2021, doi: 10.1007/s11416-021-00380-4.

[19] G. Davies, "Shining a light on policing of the dark web: An analysis of U.K. investigatory powers," *J. Criminal Law*, vol. 84, no. 5, pp. 407–426, Oct. 2020, doi: 10.1177/0022018320952557.

[20] M. Delong, B. David, and E. Filiol, "Detection of crawler traps: Formalization and implementation defeating protection on internet and on the TOR network," in *Proc. 6th Int. Conf. Inf. Syst. Secur. Privacy*, 2020, pp. 775–783. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85083024838&partnerID=40&md5=fbeb0e86415687386c466e0bb82ab7d3

[21] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proc. 13th USENIX Secur. Symp.*, Aug. 2004.

[22] Europol. *Internet Organised Crime Threat Assessment (IOCTA) 2020*. Accessed: Apr. 3, 2023. [Online]. Available: https://www.europol.europa.eu/publications-events/main-reports/internet-organised-crime-threat-assessment-iocta-2020

[23] M. Faizan and R. A. Khan, "A two-step dimensionality reduction scheme for dark web text classification," in *Advances in Intelligent Systems and Computing*. Springer Singapore, 2020, pp. 303–312, doi: 10.1007/978-981-15-1518-725.

[24] P. L. Frana, "Before the web there was gopher," *IEEE Ann. Hist. Comput.*, vol. 26, no. 1, pp. 20–41, Jan. 2004, doi: 10.1109/MAHC.2004.1278848.

[25] T. Fu, A. Abbasi, and H. Chen, "A focused crawler for dark web forums," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 61, no. 6, pp. 1213–1231, 2010, doi: 10.1002/asi.21323.

[26] K. Furumoto, M. Umizaki, A. Fujita, T. Nagata, T. Takahashi, and D. Inoue, "Extracting threat intelligence related IoT botnet from latest dark web data collection," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData) IEEE Congr. Cybermatics (Cybermatics)*, Dec. 2021, pp. 138–145, doi: 10.1109/ithings-greencom-cpscom-smartdata-cybermatics53846.2021.00034.

[27] S. Ghosh, P. Porras, V. Yegneswaran, K. Nitz, and A. Das, "ATOL: A framework for automated analysis and categorization of the dark web ecosystem," in *Proc. Workshops 31st AAAI Conf. Artif. Intell.*, Mar. 2017, pp. 170–178, cited By 5. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85046091132&partnerID=40&md5=97c23712172301caf6a2182866596ed2

[28] G. Gou, Q. Bai, G. Xiong, and Z. Li, "Discovering abnormal behaviors via HTTP header fields measurement," *Concurrency Comput., Pract. Exper.*, vol. 29, no. 20, Aug. 2016, Art. no. e3926, doi: 10.1002/cpe.3926.

[29] Y. Gupta, A. Saini, A. Saxena, and A. Sharan, "Fuzzy logic based similarity measure for information retrieval system performance improvement," in *Proc. Int. Conf. Distrib. Comput. Internet Technol.*, vol. 8337, 2014, pp. 224–232. https://www.scopus.com/inward/record.uri?eid=2-s2.0-84958551881&doi=10.1007%2f978-3-319-04483-523&partnerID=40&md5=4f33184741d7c985b77f0a84e38dd7f4

[30] D. Hayes, F. Cappa, and J. Cardon, "A framework for more effective dark web marketplace investigations," *Information*, vol. 9, no. 8, p. 186, Jul. 2018, doi: 10.3390/info9080186.

[31] D. Hayes, F. Cappa, and J. Cardon, "A framework for more effective dark web marketplace investigations," *Inf.*, vol. 9, no. 8, p. 186, Jul. 2018, doi: 10.3390/.

[32] A. Heydon and M. Najork, "Mercator: A scalable, extensible web crawler," *World Wide Web*, vol. 2, no. 4, pp. 219–229, 1999, doi: 10.1023/A:1019213109274.

[33] S. Hovrätt. (2022). *Fällande Dom I Flugsvamp 2.0-Mälet*. Accessed: Jul. 5, 2022. [Online]. Available: https://www.domstol.se/nyheter/2022/06/fallande-dom-i-flugsvamp-2.0-malet/

[34] C.-Y. Huang and H. Chang, "GeoWeb crawler: An extensible and scalable web crawling framework for discovering geospatial web resources," *ISPRS Int. J. Geo-Inf.*, vol. 5, no. 8, p. 136, Aug. 2016, doi: 10.3390/ijgi5080136.

[35] Hunchly. (2022). *Support*. Accessed: May 22, 2022. [Online]. Available: https://hunch.ly//#support-faqs

[36] IETF. (1996). *HTTP/1.0*. Accessed: Aug. 8. 2022. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc1945

[37] IETF. (1999). *HTTP/1.1*. Accessed: Jun. 8, 2022. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc2616#section-5

[38] IETF. (2022). *HTTP/3*. Accessed: Aug. 8, 2022. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc9114

[39] C. Iliou, G. Kalpakis, T. Tsikrika, S. Vrochidis, and I. Kompatsiaris, "Hybrid focused crawling for homemade explosives discovery on surface and dark web," in *Proc. 11th Int. Conf. Availability, Rel. Secur. (ARES)*, Aug. 2016, pp. 229–234, doi: 10.1109/ares.2016.66.

[40] C. Iliou, G. Kalpakis, T. Tsikrika, S. Vrochidis, and I. Kompatsiaris, "Hybrid focused crawling on the surface and the dark web," *EURASIP J. Inf. Secur.*, vol. 2017, no. 1, pp. 1–13, Jul. 2017, doi: 10.1186/s13635-017-0064-5.

[41] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt, "A critical evaluation of website fingerprinting attacks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2014, pp. 263–274, doi: 10.1145/2660267.2660368.

[42] G. Kalpakis, T. Tsikrika, C. Iliou, T. Mironidis, S. Vrochidis, J. Middleton, U. Williamson, and I. Kompatsiaris, "Interactive discovery and retrieval of web resources containing home made explosive recipes," in *Proc. Int. Conf. Hum. Aspects Inf. Secur., Privacy, Trust*. Cham, Switzerland: Springer, 2016, pp. 221–233, doi: 10.1007/978-3-319-39381-020.

[43] E. G. Kana, S. Schmidt, and R. A. Kenfack, "A web-enabled software for real-time biogas fermentation monitoring –assessment of dark fermentations for correlations between medium conductivity and biohydrogen evolution," *Int. J. Hydrogen Energy*, vol. 38, no. 25, pp. 10235–10244, Aug. 2013, doi: 10.1016j.ijhydene.2013.06.019.

[44] Y. Kawaguchi, A. Yamada, and S. Ozawa, "AI web-contents analyzer for monitoring underground marketplace," in *Neural Information Processing*. Cham, Switzerland: Springer, 2017, pp. 888–896, doi: 10.1007/978-3-319-70139-490.

[45] B. Kitchenham, *Procedures for Performing Systematic Reviews*, vol. 33, no. 2004. Keele, U.K.: Keele Univ., 2004, pp. 1–26.

[46] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Keele Univ., Keele, U.K., Tech. Rep. EBSE-2007-01, 2007.

[47] P. Koloveas, T. Chantzios, C. Tryfonopoulos, and S. Skiadopoulos, "A crawler architecture for harvesting the clear, social, and dark web for IoT-related cyber-threat intelligence," in *Proc. IEEE World Congr. Services (SERVICES)*, Jul. 2019, pp. 3–8, doi: 10.1109/SERVICES.2019.00016.

[48] P. Koloveas, T. Chantzios, S. Alevizopoulou, S. Skiadopoulos, and C. Tryfonopoulos, "InTIME: A machine learning-based framework for gathering and leveraging web data to cyber-threat intelligence," *Electronics*, vol. 10, no. 7, p. 818, Mar. 2021, doi: 10.3390/electronics10070818.

[49] M. Kumar, R. Bhatia, and D. Rattan, "A survey of web crawlers for information retrieval," *WIREs Data Mining Knowl. Discovery*, vol. 7, no. 6, Nov. 2017, Art. no. e1218, doi: 10.1002/widm.1218.

[50] J. Lee, Y. Hong, H. Kwon, and J. Hur, "Shedding light on dark Korea: An in-depth analysis and profiling of the dark web in Korea," in *Information Security Applications*. Cham, Switzerland: Springer, 2020, pp. 357–369, doi: 10.1007/978-3-030-39303-827.

[51] A. I. Levin and I. A. Voronov, "The investigation of the possibility of automated collection of information in the hidden segment of the internet," in *Proc. IEEE Conf. Russian Young Researchers Electr. Electron. Eng. (EIConRus)*, Jan. 2018, pp. 66–68, doi: 10.1109/eiconrus.2018.8317031.

[52] R. Li, S. Chen, J. Yang, and E. Luo, "Edge-based detection and classification of malicious contents in Tor darknet using machine learning," *Mobile Inf. Syst.*, vol. 2021, pp. 1–13, Nov. 2021, doi: 10.155/2021/8072779.

[53] B. Light, P. Mitchell, and P. Wikström, "Big data, method and the ethics of location: A case study of a hookup app for men who have sex with men," *Social Media Soc.*, vol. 4, no. 2, Apr. 2018, Art. no. 205630511876829, doi: 10.1177/2056305118768299.

[54] X. Liu, Q. Liu, X. Wang, and Z. Jia, "Fingerprinting Web browser for tracing anonymous web attackers," in *Proc. IEEE 1st Int. Conf. Data Sci. Cyberspace (DSC)*, Jun. 2016, pp. 222–229, doi: 10.1109/dsc.2016.78.

[55] U. Lokala, F. R. Lamy, R. Daniulaityte, A. Sheth, R. W. Nahhas, J. I. Roden, S. Yadav, and R. G. Carlson, "Global trends, local harms: Availability of fentanyl-type drugs on the dark web and accidental overdoses in Ohio," *Comput. Math. Org. Theory*, vol. 25, no. 1, pp. 48–59, Oct. 2018, doi: 10.1007/s10588-018-09283-0.

[56] *Envolve Forensics LTD. Products*. Accessed: May 22, 2022. [Online]. Available: https://en.fawproject.com/products/

[57] M. Yadav and N. Goyal, "Comparison of open source crawlers—A review," *Int. J. Sci. Eng. Res.*, vol. 6, no. 9, pp. 1544–1551, 2015. [Online]. Available: https://www.ijser.org/researchpaper/Comparison-of-Open-Source-Crawlers–A-Review.pdf

[58] S. M. M. Monterrubio, J. E. A. Naranjo, L. I. B. Lopez, and A. L. V. Caraguay, "Black widow crawler for TOR network to search for criminal patterns," in *Proc. 2nd Int. Conf. Inf. Syst. Softw. Technol. (ICI2ST)*, Mar. 2021, pp. 108–113, doi: 10.1109/ici2st51859.2021.00023.

[59] V. S. Murov and A. V. Arzhskov, "Vulnerability research onion sites TOR," in *Proc. IEEE Conf. Russian Young Researchers Elect. Electron. Eng. (EIConRus)*, Moscow, Russia, 2020, pp. 423–425, doi: 10.1109/EIConRus49466.2020.9039300.

[60] in *Proc. 12th Int. Conf. Secur., Privacy, Anonymity Comput., Commun., Storage (SpaCCS)*. Atlanta, GA, USA: Springer-Verlag, 2019.

[61] in *Proc. AHFE Int. Conf. Hum. Factors Cybersecur.* FL, USA: Springer-Verlag, 2018.

[62] in *Proc. IEEE Conf. Dependable Secure Comput.* Kaohsiung, Taiwan: IEEE, 2018.

[63] *Financial Cryptography and Data Security—FC 2011 Workshops, RLCPS and WECSR 2011, Revised Selected Papers*. St. Lucia, QLD, Australia: Elsevier, 2012.

[64] M. Nouwens, I. Liccardi, M. Veale, D. Karger, and L. Kagal, ''Dark patterns after the GDPR: Scraping consent pop-ups and demonstrating their influence,'' in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, Apr. 2020, pp. 1–13, doi: 10.1145/3313831.3376321.

[65] Juha Nurmi and Individual Contributors. (2021). *Ahmia Crawler*. [Online]. Available: https://github.com/ahmia/ahmia-crawler

[66] OSIRT. (2022). *Support*. Accessed: May 22, 2022. [Online]. Available: https://www.osirtbrowser.com/

[67] M. Pannu, I. Kay, and D. Harris, ''Using dark web crawler to uncover suspicious and malicious websites,'' in *Advances in Intelligent Systems and Computing*. Cham, Switzerland: Springer, Jun. 2018, pp. 108–115, doi: 10.1007/978-3-319-94782-2_11.

[68] J. Park, H. Mun, and Y. Lee, ''Improving Tor hidden service crawler performance,'' in *Proc. IEEE Conf. Depend. Secure Comput. (DSC)*, Dec. 2018, pp. 1–8, doi: 10.1109/desec.2018.8625103.

[69] E. Perjons and P. Johannesson, *An Introduction to Design Science*. Cham, Switzerland: Springer, 2014.

[70] K. Pham, A. Santos, and J. Freire, ''Understanding website behavior based on user agent,'' in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2016, pp. 1053–1056, doi: 10.1145/F2911451.2914757.

[71] O. Popov, J. Bergman, and C. Valassi, ''A framework for a forensically sound harvesting the dark web,'' in *Proc. Central Eur. Cybersecurity Conf.*, Nov. 2018, pp. 1–7, doi: 10.1145/3277570.3277584.

[72] K. Porter, ''Analyzing the DarkNetMarkets subreddit for evolutions of tools and trends using LDA topic modeling,'' *Digital Invest.*, vol. 26, pp. S87–S97, Jul. 2018, doi: 10.1016/j.diin.2018.04.023. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85068692166&doi=10.1016%2fj.diin.2018.04.023&partnerID=40&md5=d67e695593797d5d78d299c62ee69275

[73] R. S. Portnoff, S. Afroz, G. Durrett, J. K. Kummerfeld, T. Berg-Kirkpatrick, D. McCoy, K. Levchenko, and V. Paxson, ''Tools for automated analysis of cybercriminal markets,'' in *Proc. Int. World Wide Web Conf. Committee (IW3C2)*, 2017, pp. 1–5, doi: 10.1145/3038912.3052600.

[74] Selenium Project. (2022). *Limitations of Scaling Up Tests in Selenium 2*. Accessed: Sep. 19, 2022. [Online]. Available: https://www.selenium.dev/documentation/legacy/selenium2/parallelexecution/#running-parallel-selenium2

[75] Tor Project. (2022). *Javascript | Tor Project | Support*. Accessed: Jul. 5, 2022. [Online]. Available: https://support.torproject.org/glossary/javascript/

[76] J. T. Rabaut. (2017). *Complaint: United states of America V. Alexandre Cazes*. Accessed: Jul. 5, 2022. [Online]. Available: https://www.justice.gov/opa/press-release/file/982821/download

[77] I. G. S. Rahayuda and N. P. L. Santiari, ''Crawling and cluster hidden web using crawler framework and fuzzy-KNN,'' in *Proc. 5th Int. Conf. Cyber IT Service Manage. (CITSM)*, Aug. 2017, pp. 1–7, doi: 10.1109/citsm.2017.8089225.

[78] M. Al-Ramahi, I. Alsmadi, and J. Davenport, ''Exploring hackers assets,'' in *Proc. 7th Symp. Hot Topics Sci. Secur.*, Aug. 2020, pp. 1–4, doi: 10.1145/3384217.3385619.

[79] R. Rawat, A. S. Rajawat, V. Mahor, R. N. Shaw, and A. Ghosh, ''Dark web—Onion hidden service discovery and crawling for profiling morphing, unstructured crime and vulnerabilities prediction,'' in *Innovations in Electrical and Electronic Engineering*. Singapore: Springer, 2021, pp. 717–734, doi: 10.1007/978-981-16-0749-357.

[80] A. Razaque, B. Valiyev, B. Alotaibi, M. Alotaibi, S. Amanzholova, and A. Alotaibi, ''Influence of COVID-19 epidemic on dark web contents,'' *Electronics*, vol. 10, no. 22, p. 2744, Nov. 2021, doi: 10.3390/electronics10222744.

[81] G. Roja, N. Tulasi Chitra, K. Pushpa Rani, and B. Dhanalaxmi, ''A credit card fake detection system using image cryptography,'' *Int. J. Recent Technol. Eng.*, vol. 7, no. 6, pp. 118–122, 2019. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85067982842&partnerID=40&md5=1731fb681f03efec95814a4bd5bde7a4

[82] G. Rossi, ''Halo shapes, initial shear field, and cosmic Web,'' *J. Phys., Conf. Ser.*, vol. 484, Mar. 2014, Art. no. 012049, doi: 10.1088/1742-6596/484/1/012049.

[83] T. Sabbah, A. Selamat, M. H. Selamat, R. Ibrahim, and H. Fujita, ''Hybridized term-weighting method for dark web classification,'' *Neurocomputing*, vol. 173, pp. 1908–1926, Jan. 2016, doi: 10.1016/j.neucom.2015.09.063.

[84] Scopus. (2022). *Scopus Source List*. [Online]. Available: https://www.elsevier.com/?a=91122

[85] R. Scrivens, T. Gaudette, G. Davies, and R. Frank, ''Searching for extremist content online using the dark crawler and sentiment analysis,'' in *Methods Criminology Criminal Justice Research*. Bingley, U.K.: Emerald Publishing Limited, Aug. 2019, pp. 179–194, doi: 10.1108/s1521-613620190000024016.

[86] S. Shiaeles, N. Kolokotronis, and E. Bellini, ''IoT vulnerability data crawling and analysis,'' in *Proc. IEEE World Congr. Services (SERVICES)*, Jul. 2019, pp. 78–83, doi: 10.1109/services.2019.00028.

[87] V. Shinde, S. Dhotre, V. Gavde, A. Dalvi, F. Kazi, and S. G. Bhirud, ''CrawlBot: A domain-specific pseudonymous crawler,'' in *Communications in Computer and Information Science*. Cham, Switzerland: Springer, 2021, pp. 89–101, doi: 10.1007/978-3-030-84842-27.

[88] R. Singh, R. Nithyanand, S. Afroz, P. Pearce, M. Tschantz, P. Gill, and V. Paxson, ''Characterizing the nature and dynamics of Tor exit blocking,'' in *Proc. USENIX Secur. Symp.*, 2017, pp. 325–341, https://www.scopus.com/inward/record.uri?eid=2-s2.0-85056378676&partnerID=40&md5=1d5c86adccea06078c655008daf814b0

[89] Tor-Project. (2022). *Can I Use Tor With a Browser Besides Tor Browser*. Accessed: Sep. 9, 2020. [Online]. Available: https://support.torproject.org/tbb/

[90] Tor-Project. (2022). *How Do I Check if My Application That Uses Socks is Leaking DNS Requests*. Accessed: Sep. 9, 2022. [Online]. Available: https://support.torproject.org/ca/misc/check-socks-dns-leaks/

[91] W3C. (2022). *Facts About W3C*. Accessed: Sep. 5, 2020. [Online]. Available: https://www.w3.org/Consortium/facts

[92] J. Williams and P. Stephens, ''Analysis of the 'open source internet research tool': A usage perspective from uk law enforcement,'' in *Human Aspects of Information Security and Assurance*, N. Clarke and S. Furnell, N. Clarke and S. Furnell, Eds. Cham, Switzerland: Springer, 2020, pp. 341–352.

[93] P. Winter, A. Edmundson, L. M. Roberts, A. Dutkowska-Żuk, M. Chetty, and N. Feamster, ''How do Tor users interact with onion services?'' in *Proc. 27th USENIX Secur. Symp. (USENIX Secur.)*. Baltimore, MD, USA: USENIX Association, Aug. 2018, pp. 411–428. [Online]. Available: https://www.usenix.org/conference/usenixsecurity18/presentation/winter

[94] A. Woodruff, P. M. Aoki, E. Brewer, P. Gauthier, and L. A. Rowe, ''An investigation of documents from the world wide web,'' *Comput. Netw. ISDN Syst.*, vol. 28, nos. 7–11, pp. 963–980, 1996. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-33750402884&doi=10.1016%2f0169-7552%2896%2900064-5&partnerID=40&md5=a7b3de0659b4c6fbd7c25327da48e9dc

[95] Y. Wu, F. Zhao, X. Chen, P. Skums, E. L. Sevigny, D. Maimon, M. Ouellet, M. H. Swahn, S. M. Strasser, M. J. Feizollahi, Y. Zhang, and G. Sekhon, ''Python scrapers for scraping cryptomarkets on Tor,'' in *Security, Privacy, and Anonymity in Computation, Communication, and Storage*. Cham, Switzerland: Springer, 2019, pp. 244–260, doi: 10.1007/978-3-030-24907-619.

[96] Y. Xu, G. Chen, J. Wu, W. Xu, and Q. Liu, ''Research on dark web monitoring crawler based on TOR,'' in *Proc. 2nd Int. Conf. Inf. Technol., Big Data Artif. Intell. (ICIBA)*, Dec. 2021, pp. 197–202, doi: 10.1109/iciba52610.2021.9687954.

[97] D. Yang and P. Thiengburanathum, ''Scalability and robustness testing for open source web crawlers,'' in *Proc. Joint Int. Conf. Digit. Arts, Media Technol. With ECTI Northern Sect. Conf. Electr., Electron., Comput. Telecommun. Eng.*, Mar. 2021, pp. 197–201, doi: 10.1109/ECTIDAMT-NCON51128.2021.9425701.

[98] L. Yang, F. Liu, J. M. Kizza, and R. K. Ege, ''Discovering topics from dark websites,'' in *Proc. IEEE Symp. Comput. Intell. Cyber Secur.*, Mar. 2009, pp. 175–179, doi: 10.1109/cicybs.2009.4925106.

[99] Y. Yang, H. Yu, L. Yang, M. Yang, L. Chen, G. Zhu, and L. Wen, ''Hadoop-based dark web threat intelligence analysis framework,'' in *Proc. IEEE 3rd Adv. Inf. Manage., Communicates, Electron. Autom. Control Conf. (IMCEC)*, Oct. 2019, pp. 116–120, doi: 10.1109/imcec46724.2019.8984106.

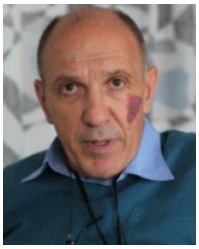
[100] Y. Yang, ''Crawling and analysis of dark network data,'' in *Proc. 6th Int. Conf. Comput. Data Eng.*, Jan. 2020, pp. 116–120, doi: 10.1145/3379247.3379272.

[101] H. Zhang and F. Zou, ''A survey of the dark web and dark market research,'' in *Proc. IEEE 6th Int. Conf. Comput. Commun. (ICCC)*, Dec. 2020, pp. 1694–1705, doi: 10.1109/iccc51575.2020.9345271.

[102] N. Zhang, M. Ebrahimi, W. Li, and H. Chen, ''Counteracting dark web text-based CAPTCHA with generative adversarial learning for proactive cyber threat intelligence,'' *ACM Trans. Manage. Inf. Syst.*, vol. 13, no. 2, pp. 1–21, Mar. 2022, doi: 10.1145/3505226.

[103] A. T. Zulkarnine, R. Frank, B. Monk, J. Mitchell, and G. Davies, ''Surfacing collaborated networks in dark Web to find illicit and criminal content,'' in *Proc. IEEE Conf. Intell. Secur. Informat. (ISI)*, Sep. 2016, pp. 109–114, doi: 10.1109/isi.2016.7745452.

JESPER BERGMAN received the B.Sc. degree in computer and systems sciences and the M.Sc. degree in information security from Stockholm University, in 2016, where he is currently pursuing the Ph.D. degree in digital forensics. After a few years in the industry, he became a Teaching and Research Assistant with the Department of Computer and Systems Sciences, Stockholm University. In the last five years, he has participated in several EU and national research projects.



OLIVER B. POPOV (Member, IEEE) received the Ph.D. degree in computer science (Artificial Intelligence) from the Missouri University of Science and Technology, Rolla, USA, in 1987. He is a Professor of computer science in the area of information security and forensics with the Department of Computer and Systems Sciences, Stockholm University. He is a Professor with the Faculty of Computer Science and Computer Engineering, Saints Cyril and Methodius University, and with the Department of Information Technology and Media, Mid Sweden University. In the last 30 years, he has participated in more than 50 international research projects, and the corpus of work comprises 200 peer-reviewed publications including journal and conference articles, book chapters, and nine books.