C2Store: C2 Server Profiles at Your Fingertips

VIVEK JAIN, University of California, Riverside, USA
S M MAKSUDUL ALAM, University of California, Riverside, USA
SRIKANTH V. KRISHNAMURTHY, University of California, Riverside, USA
MICHALIS FALOUTSOS, University of California, Riverside, USA

10

How can we build a definitive capability for tracking C2 servers? Having a large-scale continuously updating capability would be essential for understanding the spatiotemporal behaviors of C2 servers and, ultimately, for helping contain botnet activities. Unfortunately, existing information from threat intelligence feeds and previous works is often limited to a specific set of botnet families or short-term data collections. Responding to this need, we present *C2Store*, an initiative to provide the most comprehensive information on C2 servers. Our work makes the following contributions: (a) we develop techniques to collect, verify, and combine C2 server addresses from five types of sources, including uncommon platforms, such as GitHub and Twitter; (b) we create an open-access annotated database of 335,967 C2 servers across 133 malware families, which supports semantically-rich and smart queries; (c) we identify surprising behaviors of C2 servers with respect to their spatiotemporal patterns and behaviors. First, we successfully mine Twitter and GitHub and identify C2 servers with a precision of 97% and 94%, respectively. Furthermore, we find that the threat feeds identify only 24% of the servers in our database, with Twitter and GitHub providing 32%. A surprising observation is the identification of 250 IP addresses, each of which hosts more than 5 C2 servers for different botnet families at the same time. Overall, we envision *C2Store* as an ongoing effort that will facilitate research by providing timely, historical, and comprehensive C2 server information by critically combining multiple sources of information.

CCS Concepts: • **Security and privacy → Malware and its mitigation**.

Additional Key Words and Phrases: C2 servers, botnet, Twitter, GitHub

ACM Reference Format:
Vivek Jain, S M Maksudul Alam, Srikanth V. Krishnamurthy, and Michalis Faloutsos. 2023. C2Store: C2 Server Profiles at Your Fingertips. *Proc. ACM Netw.* 1, CoNEXT3, Article 10 (December 2023), 21 pages. https://doi.org/10.1145/3629132

1 INTRODUCTION

Identifying Command and Control (C2) servers is an essential step in mitigating the damage caused by botnets. C2 servers are the "operational headquarters" of botnets, which allows attackers to remotely control compromised devices and command them to launch cyberattacks. C2 servers could be identified by an IP address or domain name and their family corresponds to the botnet they control. Identifying C2 servers is a critical component in any defense against botnets, as it can help us neutralize them and even take over their botnet [54, 66]. Due to their significance, C2 servers are essential Indicators of Compromise (IoC). Consequently, one would expect that the security

Authors' addresses: Vivek Jain, vjain014@ucr.edu, University of California, Riverside, USA; S M Maksudul Alam, salam031@ucr.edu, University of California, Riverside, USA; Srikanth V. Krishnamurthy, krish@cs.ucr.edu, University of California, Riverside, USA; Michalis Faloutsos, michalis@cs.ucr.edu, University of California, Riverside, USA.

2834-5509/2023/12-ART10
https://doi.org/10.1145/3629132

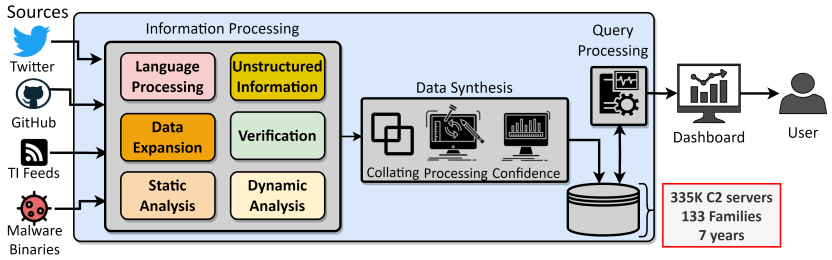


Fig. 1. *C2Store* collects data from multiple sources and synthesizes the information into a queryable format.

community would have a well-established comprehensive mechanism to identify and track C2 server profiles[1]. However, such a comprehensive reference source does not seem to exist currently.

Problem: *How can we create a definitive capability for tracking C2 servers?* The requirement is to collect, filter, and combine sources of information in order to generate a continuously updating comprehensive archive. We use the term *definitive* to characterize a capability that provides: (a) the maximum possible number of C2 servers, (b) representatives from many botnet malware families, (c) historical data pertaining to these servers, (d) attribution and associated level of confidence, and (e) continuous updates. Currently, there are several fragmented sources of information that can be loosely categorized into: (i) threat feeds, (ii) research studies, and (iii) security reports. The available information is too widely dispersed and unstructured to provide any holistic view. The challenges in building such a capability include: (a) extracting C2 information from multiple sources, with each source introducing different challenges; (b) validating the collected data to ensure high confidence, and (c) enabling powerful and informative queries and visualization. For example, we would like to be able to ask "complex" queries, such as: "Find all malware families associated with C2 servers hosted on AS 45080 reported after 2022".

A missed opportunity: With respect to sources of C2 server information, we identify an untapped type of source: information shared by experts on social media. In the spirit of collaboration, many security experts share timely information about emerging threats (e.g., C2 servers and malware binaries and their hashes) on platforms like Twitter or GitHub. As one may expect, the challenge is to identify the right people and places from whom and where to gather this information.

Previous work: To the best of our knowledge, there does not seem to be an effort that provides comprehensive open-access C2 server information. Prior efforts can be grouped into the following categories. First, several threat feeds [11, 20, 83] provide C2 server information. However, these feeds often lack complete C2 server profiles, providing only C2 addresses as blacklists, are not always timely, or lack explanations on how the information was collected or validated. Second, several efforts utilize (botnet) malware binaries to analyze specific families [13, 44] or different phases of their lifecycle [10, 12, 43, 58] or to find live C2 servers by actively scanning an IP space [29, 30] or by using tools like Shodan [64, 78]. The challenge lies in the computational complexity of creating sandboxes to activate the binaries, especially for a large-scale study. Third, some studies [27, 45, 74] focus on mining GitHub repositories and other public archives but have completely different goals, such as finding repositories with malware source code or vulnerabilities. Fourth, several studies [19, 31, 41, 42, 48, 57] aim to detect C2 traffic by analyzing network traces. We further discuss related work in §8.

Contributions: Our overarching contribution is *C2Store*, which we want to establish as the definitive open-access C2 server capability. From a quantitative point of view, our capability can be described by the following numbers: (a) 335,967 C2 servers, (b) five types of sources with 135 distinct

[1]We define a C2 server profile as the C2 address along with comprehensive metadata such as activeness and liveness over time, their family, the AS/ISP that hosts it, and C2-bot packet traces. For brevity, we refer to C2 server profiles as C2 servers.

sources, (c) 133 malware families, and (d) spanning 7 years. To obtain the above numbers, our work makes the following contributions. First, we develop methods to mine non-traditional sources (GitHub, Twitter) and show the value of such an effort. Second, we create, arguably, the largest open-access database of C2 server profiles annotated with: (a) the botnet family type, including ports and even packet traces, when available, (b) spatiotemporal information, such as the time of the first and most recent reports. Third, we enable powerful features by supporting semantically rich queries and powerful visualizations.

We depict an overview of *C2Store*, which consists of two main modules: (a) Information Collection and (b) Information Synthesis. The goal is to have *C2Store* periodically gather data from various sources in Figure 1. The Information Processing module handles the challenges associated with processing information from these sources. The Information Synthesis module combines the C2 server information from the different sources and incorporates additional useful metadata as discussed in §3.2. Finally, our capability supports queries and informative reports, and visualizations, transforming the raw data into actionable intelligence that is both user-friendly and easily accessible. The following key findings provide strong evidence of the value of our approach:

a. High precision methods: We show that we can identify C2 servers by mining Twitter and GitHub with high precision (97% and 94%, respectively).

b. Non-obvious sources provide significant information: We find that the most popular 6 threat feeds contribute only 24% of the C2 servers in our database. Our Twitter and GitHub mining methods provide 32% of the servers; the remaining comes from the malware binary analysis.

As proof of its usefulness, we use *C2Store* to study the spatiotemporal and behavioral patterns of the C2 server. Indicatively, we provide some highlights of our observations (details in §6).

a. Multi-family hosting: We identify **250 servers running five distinct families of C2 servers at the same time on different ports**, including Sliver, Cobalt Strike, Mythic, and Deimos. This observation can point to collaborations across botnet malware families or professionally streamlined "C2 server-hosting" services.

b. Strong spatial locality: The placement of C2 servers exhibits spatial locality, i.e., there exist **family-specific hotspots of C2 servers**. For example, we find that 21% of RedLine Stealers' C2 servers are situated within a few Autonomous systems (ASes) located in The Netherlands, while 20% of Cobalt Strike C2 servers are within AS 45080. This observation can be used by efforts that want to scan IP spaces to detect live servers.

c. Hiding C2 server communications via pastebin.com: We find that **1306 malware binaries** use pastebin.com for their C2 communication, a popular online text-sharing platform. Note that we reported this phenomenon to the administrators of the website. (details in §6.3.2).

d. The "undead" botnet malware: We find that 10-year-old malware families, including Mirai (2016) and Cobalt Strike (2014), remain at the top of the list of active malware in our database. This holds true even if we focus on reported active servers within the last two years (see §4.2).

Open-sourcing for maximal impact: We intend to make our datasets and tools available[2] for research purposes. These artifacts include Twitter accounts, GitHub repositories, scripts, and tools. In fact, one of the motivations behind this work is the lack of a community-wide reference capability focusing on C2 servers and providing semantically rich metadata. In addition, we provide user-facing functionalities (https://c2store.github.io/) that will allow users to extract information efficiently.

[2]Although the access to the community will be open, we will employ mechanisms to share information only with carefully vetted individuals, which would include researchers and security professionals.

2 BACKGROUND AND CHALLENGES

In this section, we first provide background on the information available on various platforms. Then, we discuss various challenges in extracting data from these platforms.

A. Terms and concepts: A **malware binary** is an executable program that runs on a device, transforming it into a bot. It is associated with a specific malware family. The term **malware family** refers to a class of malware that share code and have similar functionality, including the communication protocol with a C2 server. A **C2 server** is an operational headquarter that controls the botnet of a specific family. The family of the C2 server corresponds to the family of the botnet it controls. A **botnet** refers to an independently controlled army of bots typically of the same family.

B. Our novel sources of information: We briefly outline the sources of information that we use when collecting information on C2 servers.

a. Malware binaries. Within the malware binary, embedded artifacts such as IP addresses or domain names can reveal the presence of a C2 server. In addition, when bots execute the malware binary, they also try to communicate with their respective C2 server and, in turn, reveal the server identities. These binaries are often released by the community and can be downloaded from platforms like VirusTotal (VT) [8], MalwareBazaar [62], and VirusShare [87].

b. Threat feeds. Among other data, threat feeds like ThreatFox [83] and Fedotracker [1] provide C2 information in formats like CSV or JSON. These feeds may be offered by commercial entities, non-profit organizations, or both [23, 60].

c. Security platforms. Online services like VT, MalwareBazaar, and VirusShare allow users to scan unknown binaries and IP addresses to assess the potential maliciousness from multiple antivirus (AV) engines and other security tools. Threat analysts and users of these platforms often post additional information about malware binaries, such as C2 servers, Yara rules [70], and syscall traces for a binary within the platform's comment section.

We use comments left by users on security platforms in a creative way. We identify that this information has twofold benefits: (i) it enables us to identify and confirm C2 servers associated with the malware binary, and (ii) we are able to identify active users who frequently post relevant information. This untapped information holds great importance; we utilize these users' profiles and their relevant connections on Twitter and GitHub to mine C2 information that they post publicly.

d. GitHub [3] is a software repository and collaboration platform. Users create repositories to store and share their code, data, and documents, enabling an active collaborative culture.

e. Twitter. There is hardly a need to explain what Twitter is [16, 55]. A tweet post is associated with the following fields, which we will use in our study: (a) text information, (b) time, (c) location, and (d) community response by the number of likes and comments. Twitter user profiles can provide information such as the user's bio, followers, following, and timeline of tweets.

Extracting C2 server information from social media comes with significant challenges.

Challenge 1. The needle in the haystack: The first significant challenge is the sheer volume of data generated on these platforms. Finding and extracting the desired information is not straightforward: Searching for specific keywords or hashtags may not always yield relevant results.

Challenge 2. Language ambiguity: Extracting useful information from unstructured, user-generated information is also non-trivial. In addition, the use of abbreviations and jargon further exacerbates the language complexity issue.

Challenge 3. Correctness and timeliness: Ensuring the validity of the extracted information is non-trivial. In both Twitter and GitHub, anyone can create an account and share arbitrary (mis)information. The timeliness of reporting is an aspect of the validity of the information.

3 METHODOLOGY

We describe the methodology that we use in our *C2Store* capability, consisting of two phases. We leverage some well-established approaches from prior works and tailor them to our purposes. While some of these techniques have been explored in the past in different contexts, our contribution lies in customizing these for the needs of our effort and in combining them into an efficient complete capability. Ultimately, the value of our effort lies in providing a comprehensive measurement-based understanding of C2 servers, which we will make available to the research community at large.

3.1 The Information Collection Phase

In this first phase, we gather a diverse set of C2 addresses, including IP addresses and domain names, from a variety of sources. Some of the collection capabilities that we develop are novel and tap into sources that are not traditionally used in similar studies.

3.1.1 Threat intelligence, public blacklists and reputation feeds. We obtain C2 information from various threat feeds, listed in Table 1, which offer C2 addresses as IoC. However, these feeds have certain limitations. First, each feed seems to cover a limited number of malware families [23, 60]. Second, each feed presents the information with varying contextual information and different formats. Third, the information provided by these feeds is not always up-to-date; we have observed delays between when the information was generated and when it was incorporated into such feeds (details in §4).

Table 1. C2 servers obtained per threat feed.

Feeds	#C2s
AlienVault [11]	27,434
FedoTracker [1]	6,103
blocklist.de [9]	8,307
Viriback [20]	8,846
ThreatFox [83]	79,771
Becknow [18]	12,208
Total	142,669

3.1.2 Malware binaries. We collect over 250,000 botnet-specific malware binaries involving C2 communication from sources like VT, VirusShare, and MalwareBazaar. We also consider several malware samples from previous works [10, 12, 30, 80], which were kind enough to share their datasets. We only select those binaries that are deemed malicious by at least 5 AV engines in VT. The threshold 5 is aligned with best practices [91]. It turns out that the coverage of our dataset is extensive: we have binaries for various platforms, including but not limited to Windows, Linux, and IoT malware, designed for various architectures such as Renesas SH, Motorola 68000, SPARC, Intel 80386, ARM, PowerPC, MIPS, ARC Cores Tangent-A5, and AMD x86-64.

We perform ***static*** and ***dynamic*** analysis on the collected malware binaries, as these two approaches complement each other. ***Static analysis*** is beneficial when the binary fails to activate in a sandbox environment. On the other hand, ***dynamic analysis*** proves valuable when binaries are packed or obfuscated, preventing meaningful results from being obtained through static analysis alone. We follow methodology from previous works [10, 12, 28, 58, 65, 73, 91] and extend them with the purpose of extracting the corresponding C2 servers from the malware binaries.

a. Using Static Analysis: First, we check if any binary packers are used to obfuscate the binaries from detection. We developed an extensible tool by utilizing standard utilities such as `file`, `hexdump`, and UPX [63] for detecting and unpacking the malware binaries. We make available all of our tools and datasets in our GitHub repository. Then, we use `strings`[3] to extract IP addresses and domains that are embedded inside the malware binaries. We use `tldextract` [51] when extracting domains to obtain correctly formatted domain names.

b. Using Dynamic Analysis: We execute each malware sample in an architecture-specific virtual machine (sandbox) for 60 seconds and monitor the system call traces using tools like `strace` and capture network traces. Recognizing that the execution time can impact observed behavior [61],

[3] *strings* is a command-line tool that extracts printable character strings from binary files, which can be useful in identifying embedded IP addresses, domains, and other text-based information within malware binaries.

Table 2. The C2 servers from malware binaries using static and dynamic analysis and from other datasets.

Source	Binaries		IoC (IP and domains)		
	Total	Activated	Static	Dynamic	Total
Mal. Bazaar	56,570	34,508	36,864	32,776	39,734
VirusTotal	65,931	41,714	42,567	39,138	58,684
VirusShare	12,684	8,467	8,235	6,456	9,112
Other Datasets	-	-	-	-	189,433

Note: We directly report IoC for other Datasets [10, 12, 30, 80] after analyzing their released dataset. Some of these may contain duplicates and will be filtered after Synthesis Phase discussed in §3.2.

we address this limitation by implementing a new dummy sleep function with empty functionality and inject it using `LD_PRELOAD` [71] environment variable when launching the binary. This helps us bypass any delay the original sleep function introduces. We follow the standard criterion [12] for considering the activation of malware as being successful. First, we ascertain if the binary creates three or more processes in the sandbox. Second, we check if it invokes 100 or more system calls. Upon completing the execution, we proceed to examine the network traffic generated by the malware. Utilizing the sandbox capabilities of CnCHunter [29], we extract C2-bound traffic with a reported 92% precision.

We show the outcomes of our static and dynamic analysis in Table 2.

3.1.3 Twitter. Our approach to extract C2 server information from Twitter follows four steps: (a) we identify an initial set of reliable users, (b) we find the Twitter accounts of these users, (c) we expand the set of Twitter users that share C2 information, and (d) we fetch the tweets and extract the C2 server-related information. We describe these steps in more detail below.

a. Identify the initial set of reliable users: To find reliable users, we identify frequent uploaders of malware binaries and individuals who post informative comments in VirusTotal and MalwareBazaar (discussed in §2). To ensure reliability, we used VirusTotal stars [88] awarded by the VirusTotal community to filter out low-confidence users.

b. Finding reliable users on Twitter: After identifying reliable users, we conducted a search to locate these individuals on Twitter. Leveraging recent studies that indicate threat actors use consistent usernames across security forums, we applied a similar approach to find relevant threat analysts. Using a username-matching technique, we successfully located most of these users on Twitter. Our approach was guided by two hypotheses: (i) these users would share similar information on their social media accounts, and (ii) they may retweet or have connections with other individuals who share similar, albeit distinct, information. We validated our hypotheses with manual inspection.

c. Expanding the set of Twitter users: To find more trusted users of interest on Twitter, we start by considering the reliable users we found in ***step (a)***. In addition, we manually selected the Twitter accounts of security companies. We create a directed graph of their connections (following and follower) and mark these initial users as trusted nodes. We grow the graph up to two degrees (being conservative) of their connections (i.e., neighbors of neighbors), mainly to capture the most relevant users. In addition, we assign node weights based on the engagement of their tweets, such as comments, retweets, and likes from trusted users. This allowed us to take into consideration their level of interaction.

Identifying additional reliable users: To classify other nodes in the graph as trusted, we build on the concept of a popular (or trusted) account. A popular account intuitively follows many other popular accounts and has a higher probability of being reached by walking randomly in the graph. The PageRank algorithm [24], commonly used in search engines, provides a metric for measuring popularity. Using [21, 53], we discovered several communities, such as security analysts, threat intelligence, and software engineers. This enables us to identify users who are likely to have a high

level of trust and reputation in the cybersecurity community. Our method provides a systematic approach to identifying reliable trust and reputation associations on Twitter, which is especially crucial in the cybersecurity industry, where trust and reputation are fundamental factors.

Graph Analysis: We use `twitter-graph` [56] to fetch the connections and Gephi [17] to perform various clustering analyses such as modularity, spectral partitioning, and PageRank. For example, we show how using a highly reputed VT user reveals more relevant connections. The 2-level connection graph of his Twitter connections is shown in Figure 2. The size and color of a bubble represent the measure of popularity (trust and reputation) in the users' community. By selecting large and dark bubbles, *C2Store* uncovered additional trusted accounts involved in posting or engaging with C2-related content. We conducted a meticulous review of the content posted by these accounts to validate their relevance.

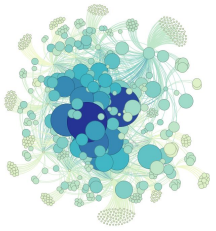


Fig. 2. Network of experts: the connection graph of Twitter users posting C2 information.

Overall, by implementing this technique, we were able to find more than 150 additional trusted threat analysts just by having 3 high-rated users initially found on VT that otherwise would be difficult to track down.

Avoiding Data Poisoning: To address the concerns regarding data poisoning, we meticulously chose 70 users by manual inspection and mined their posted tweets to extract C2 information. The manual selection processes are designed to identify and exclude potentially misleading information. This selection was primarily driven by the limitations imposed by the Twitter API, which restricts fetching a large amount of data. We discuss in the Discussion section (§7) about potential future work regarding automated identification for the bot or fake accounts.

d. Extracting C2 information from tweets: We utilize the `twarc2` [7] tool to fetch tweets from these individuals and search for information regarding C2 servers. Although we use this tool, extracting C2 information from tweets poses additional challenges. First, tweets are subject to various language complexities discussed in §2, and users may not necessarily post the information in a fixed format. To overcome these challenges, we utilize ChatGPT[4], the state-of-the-art natural language processing (NLP) model. With the help of ChatGPT, we are able to extract the C2 information posted even in different languages and dialects. Secondly, C2 information may be embedded in images accompanying the tweets. To handle these cases, we extract the text from the images by using `tesseract` [5, 79], an optical character recognition engine. Combining these advanced techniques, we create our comprehensive and effective C2 extraction system that overcomes associated challenges in retrieving and assimilating relevant data from Twitter.

3.1.4 GitHub repositories. During our investigation, we discovered that some analysts and users also share GitHub repositories containing pertinent information on C2 servers. This opened up a new avenue for data collection and analysis. We identified relevant repositories from comments in VT or tweets of reputable users and crawled those repositories. Our investigation uncovered two types of repositories: (i) repositories containing code for various C2 servers, such as the Mirai [36], and (ii) repositories that serve as datastores for popular threat intelligence streams. We also used SourceFinder [74], due to its high precision and recall, to find the relevant repositories related to C2 servers. In total, we found more than 60 such repositories. We chose 40 repositories by manual selection based on their content and mined them to extract C2 information.

Extracting C2 information from GitHub repositories: We used `PyGithub` [4] to download all identified repositories of our interest. We faced similar challenges, such as Image processing and

[4]ChatGPT has been trained on a massive amount of text data, enabling it to understand and interpret the nuances of language, including syntax, grammar, and semantics.

Language complexities, in GitHub repositories to those we faced with Twitter, which we overcame using similar approaches discussed in §3.1.3. In addition, certain GitHub repositories only store data for a limited period of time. For instance, C2IntelFeeds [33] solely retains records of Cobalt strike C2 addresses detected within the past 30 days, removing any older entries. To address this challenge, we crawled through the commit history. This has three-fold benefits: (a) we could retrieve older entries; (b) we determined the date the entries were made; (c) we were able to find the list of authors who contributed to the repository, which we utilize for establishing confidence.

3.2 The Information Synthesis Phase

In the second phase, which we call the synthesis phase, we aggregate information from various sources. This multi-step process involves collating the data, filtering, and structuring them into our unified and queryable format.

Step 1. Unifying the data: In this step, we collect data from different sources and consolidate them into a unified database to facilitate further analysis. This involves collecting relevant metadata such as timestamps, author names, and platform-specific information to complete the information.

Step 2. Deduplication: This step ensures that there is no duplication of information in the dataset. In case of collision, we mark a source as the primary source when the information was reported first on that platform (who-reported-first policy).

Step 3. Filtering: This step involves removing private IP address entries, top-1000 domains from Tranco's list [68], and irrelevant repositories that do not contribute to the analysis. This step ensures that the dataset is focused on relevant information and that the analysis is based on actionable data.

Step 4. Retaining high confidence data: This particular stage holds significant importance in establishing confidence relating to each record. To accomplish this, we implement three measures. First, we identify the number of sources reporting the information; the more sources reporting the same information, the more confidence we have in the record. Second, we assess the level of interaction related to the information. For instance, if the information originates from Twitter, we consider the number of trusted users who have liked, retweeted, or commented (and their sentiments) on the tweet. Third, we leverage VT to scan and evaluate each IP/domain using multiple AV engines. To achieve greater variability, we check these C2 addresses with VT four times over a span of a month.

Step 5. Incorporating Metadata: To build a complete C2 profile and improve the utility of *C2Store*, we incorporate various types of useful metadata information:

a. Spatial information: We offer two types of spatial information. Firstly, we provide the geolocation of C2 servers. This aids in understanding their geographical distribution and potential regional patterns. Secondly, we provide details about ASes, enabling us to identify and establish the reputation of subnets commonly associated with malicious activities. To obtain this information, we leverage third-party tools such as geoIP2 [2], ipinfo [6], shodan [78], and pyasn [14].

b. Temporal information: We offer two types of temporal information. Firstly, we provide the timestamps of the first and most recent reports. Secondly, we provide data on the longevity and liveliness of C2 servers over time. This information aids in understanding their evolution, popularity, and any changes in their usage patterns over time. To obtain this information, we deployed a cron job that runs hourly. It uses masscan [39] and ZGrab [34] to discover the open ports on the C2 servers and fetch service banners[5] from network service running on those ports.

[5]**Service banners** are messages that network services, like web servers, automatically return to clients upon message exchanges (e.g., HTTP headers). It provides details like software version, server type, and additional information to clients connecting to network services. These messages can provide insights related to the software involved and potential attack vectors, aiding in incident identification, response, and mitigation.

c. Family information: To identify the family of a C2 server, we employ several techniques. First, we retrieve family labels from the original source of information, if available. Second, we utilize a comprehensive set of 1000 Yara rules sourced from VT and previous research [12]. These rules are applied to analyze malware binaries and leverage AVClass2 [76, 77] to assign them to specific families, establishing the connection between them and the corresponding C2 server family. In addition, we collect signatures from reputable sources such as NVD [22], CVE [89], and previous studies [10]. These signatures are then compared against the service banners obtained through our cron job, further aiding in identifying the C2 server family. These combined methods provide a robust approach to accurately determine the family of a given C2 server.

d. Behavioral information: We have also developed functionality to study unusual C2 server behaviors. The functionality consists of a combination of (a) active scanning and (b) data processing scripts, but their combined outcome provides significant insights, as discussed in §6.

e. Network traces: We obtained various network traces, such as C2-bot interaction and DNS resolutions, from the dynamic analysis of malware binaries (as done in §3.1.2). These network traces could be the basis for deriving network signatures and determining detection rules for firewalls.

4 C2STORE: SCOPE AND EVALUATION

We evaluate our approach, and we find that: (a) our novel methods for extracting data from Twitter and GitHub are relatively accurate, and (b) all methods provide significant contributions to the combined *C2Store* dataset. Specifically, we answer the following questions:
Q1. How accurate are our novel methods? (§4.1)
Q2. What is the contribution of each source? (§4.2)
Q3. What is the timeliness of each source? (§4.2)
Q4. How many malware families do we capture? (§4.2)

4.1 Evaluating our social mining methods

Observation 1*. Our methods can extract C2 server information with a precision of 97% from Twitter and 94% from GitHub.*

Evaluating Twitter mining method: To evaluate the quality of the C2 addresses collected from Twitter, we utilize VirusTotal[6] as a reference source. We consider an address to be malicious[7] if it is classified as such by at least five AV engines in VT. This threshold is based on established best practices [12, 30, 91], and helps increase our confidence in the classification. We find that our Twitter-based method exhibits an overall precision of 97%. In fact, the precision for C2 information posted before Jan. 2023 even increases to 99%. These results provide strong validation for our ability to: (a) select appropriate Twitter users and (b) extract information from their posts.

Evaluating GitHub mining method: We use the same approach to evaluate the quality of the data from GitHub as we did for Twitter data. We find that our GitHub-based method exhibits a precision of 94% in identifying C2 servers. Again, we use a threshold of five AV engines to confirm the maliciousness of an address.

Overall, we extend our gratitude to the users who contribute by sharing C2 information. Our efforts lie in establishing a systematic initiative on collecting and unifying the posted information, eliminating the need for users to navigate through a potentially cumbersome and bureaucratic vetting and synthesizing process to make this information accessible.

[6]VirusTotal provides information for a requested IP address, but it does not provide a public threat feed. If it does in the future, we will incorporate it in *C2Store*.
[7]The term "malicious" is associated with the activities of the entity connected to the IP address rather than the address itself.

4.2 The contribution of the sources

We present an overview of the contribution of each type of source in our current C2 server dataset.

Source	Initial Step 1	Dedup Step 2	Filter Step 3	Final Step 4
Binaries	238,746	187,578	167,821	148,003
Feeds	142,669	106,053	95,238	81,481
Twitter	69,193	57,762	56,424	54,731
GitHub	62,870	56,708	54,874	51,752
Total	513,478	408,101	374,357	***335,967***

Table 3. An overview of the data from different types of sources and our processing pipeline.

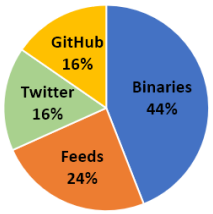


Fig. 3. The effective contribution from each source attributing a server on the source that reported it first.

Observation 2*. Our social-media-based methods provide significant information as Twitter and GitHub contribute information on 32% of the total C2 servers.*

Source type contributions. We plot the contribution of different types of sources in Figure 3 by attributing a server to the source that reports it first. A significant portion of valuable information (32%) is obtained from Twitter and GitHub. We attribute this to the ease and speed with which security experts can share information over social media. We conjecture that threat feeds either get this information later or follow a slower process in releasing the information.

We also provide a more in-depth analysis of the contributions of each type of source through our four-step synthesizing pipeline in Table 3. For each source, the table shows how many C2 addresses we found initially (step 1), how many remain after deduplication across all sources after the "who-reported-first-policy" (step 2), and filtering (step 3). The last column (step 4) shows how many C2 addresses are accounted for from that single source, illustrating the importance of particular collection methods for obtaining a diverse dataset. Due to space limitations, we highlight only a few key observations.

a. Our synthesizing process reduces the raw number of C2 servers by 35%: we start with 513K addresses, and we end up with 335K after applying deduplication, filtering, and other quality-improving synthesizing steps (mentioned in §3.2).

b. The contribution of the threat feeds is reduced by 42.96% in our pipeline from 142K to 81K. The main reduction occurs at the deduplication step when the number of servers is reduced by 26%. One of the reasons is that the threat feeds report some of the servers later than other sources, as we will discuss extensively below.

Observation 3*. Threat feeds are slower to report new C2 servers than our social media methods: 71% of new C2 servers were not reported by any threat feeds for at least 5 days.*

Timeliness of information. We study the timeliness of the information on the threat feeds in more detail. The motivation is that a significant number of servers is first reported by our social media sources. We conducted the following study. For a randomly selected day (March 27th, 2023), we identify the C2 addresses reported first by our chosen Twitter users. We obtained 298 C2 server addresses, and we checked to see when they would be reported by threat feeds. We observe that none of the feeds individually report more than 7% of the new C2 servers after 5 days, and even after 10 days, the individual coverage remains at 12% or below. If we combine all six threat feeds, we observe that nearly 71% of the addresses are not flagged as malicious for at least 5 days, while 39% are still not reported 10 days later.

Delayed validation: threat feeds corroborate our methods. The above study provides an additional, albeit delayed, validation for the correctness of our social media methods: the threat feeds report a significant number of the C2 servers that are first reported by our methods. In fact, we found

that 96% of the servers found from our social media were eventually reported by at least one of the threat feeds 25 days later in the above study.

The importance of timely information in detecting and containing botnets has been widely documented [13, 23, 44, 60]. For example, Mirai, first reported in August 2016 [32], had 600,000 IoT devices by late November and caused DDoS with a traffic volume close to 1 Tbps [15, 49, 67].

***Observation 4**. The coverage of our dataset is broad with 133 families represented.*

Family-centric observations. We wanted to assess the scope of our dataset in terms of the number of malware families. We find that our dataset includes 133 malware families, targeting a range of platforms, including Windows, Android, IoT devices, and others. We associate family information through our methods described in §3.2.

***Observation 5**. Nearly 10-year-old malware families remain among the top active botnets.*

In Figure 4, we list the top 10 malware families in terms of reported C2 servers since the beginning of 2022. First, the 10-year-old Cobalt Strike family is dominant, with roughly 37% of the newly reported servers. Similarly, variants of the Mirai malware family are prominent, although Mirai was first discovered in 2016. This suggests that old botnet families are still prevalent among threat actors. The persistence of these older families highlights the importance of regular software updates and security patches. Many IoT devices, for example, are not designed with security in mind and may not receive regular updates, leaving them vulnerable to attackers.

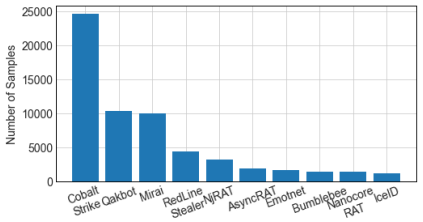


Fig. 4. The top 10 most active C2 servers by malware family since 2022.

5 C2STORE: AN ONLINE INTERACTIVE CAPABILITY

In this section, we present the interactive capabilities of *C2Store*. By leveraging widely-adopted open-source tools, we ensure the seamless maintenance of our system and prioritize enhancing the quality of our methodology for fetching relevant information about C2 servers.

a. Database: We store all collected C2 server profile records in a BigQuery database. This enables fast retrieval and processing of data, facilitating timely analysis and insights. In addition, it provides a scalable, secure, and reliable infrastructure with built-in data redundancy and failure recovery capabilities.

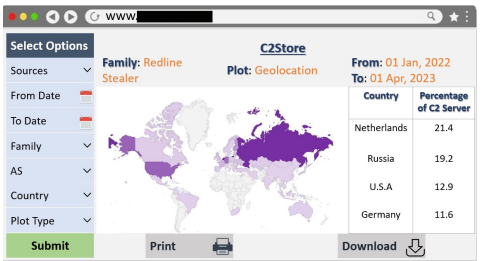


Fig. 5. Our *C2Store* platform: the geographic distribution showing hotspots for the Redline Stealer family.

b. Integration with Grafana [52]: We developed a Grafana dashboard and integrated it with our BigQuery C2 server database. It interacts with BigQuery to provide intuitive ways to explore and understand the C2 server data. Users can create interactive visualizations, charts, and graphs and write SQL queries directly within the dashboard, enabling seamless data exploration. For anonymity, we have not provided the URL to our Grafana dashboard for the review cycle but plan to release it on GitHub.

c. *C2Store* as an online platform: To amplify the usefulness, we go beyond simply making our data available. We provide an online platform that can facilitate both data access and its analysis. Our platform provides both: (a) a smart querying capability, and (b) analysis and visualization capabilities. A screenshot of our platform is shown in Figure 5. Our *C2Store* supports the following capabilities.

1. Semantically rich and complex queries. Our online monitoring dashboard makes it easy for users to perform complex queries without the need for technical expertise. With the use of filters, users can customize their queries and obtain the exact data they need.

2. Informative visualizations. Our platform offers advanced visualization capabilities, unlike traditional threat feeds that only provide CSV data dumps or a plain table. We allow users to create customizable graphs, charts, and to display trends and patterns in addition to CSV or JSON dumps.

3. Comprehensive metadata. Unlike most current available datasets, *C2Store* provides extensive annotations such as family-specific geographic distribution, the time it was reported and identified as active, the AS that hosts it, malware family, and, in some cases, the active ports, and the C2-to-bot communication as captured packet traces.

Overall, a user of our system can consider four broad types of studies which could focus on: (a) temporal activity, (b) malware family, (c) geolocation or AS, or (d) behavioral activities. In Figure 5, we show specifics of user-interaction with *C2Store*: (a) the Redline Stealer family, (b) reported between 2022-2023, (c) and reported by at least two sources for increased confidence. The geographical representation and table in Figure 5 are based on real data. Similarly, we more broadly investigate the spatiotemporal and behavioral properties of the C2 servers using *C2Store* in §6.

6 RESULTS: PATTERNS AND BEHAVIORS

This section presents our findings on the spatial and behavioral properties derived from our *C2Store* capability. For ease of exposition, we ask straightforward questions instead of queries (as we did in §5). We then present the direct results and interpret them to derive insights or key takeaways.

6.1 Spatial Analysis

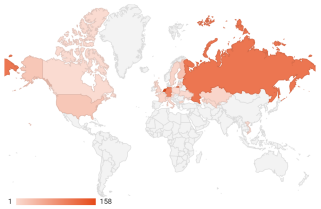
We analyze the spatial properties of C2 servers by answering the following questions:
Q1. How are the C2 servers distributed geographically? (§6.1.1)
Q2. How are the C2 servers distributed across ASes? (§6.1.2)

Observation 6*. C2 servers exhibit strong locality: the servers for a family are often clustered at a few ASes and regions. This can inform active scanning efforts to optimize the selection of IP spaces.*

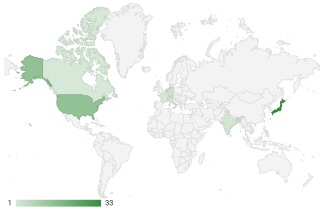
6.1.1 Geolocation properties. We first analyzed the geographical distribution of all C2 servers in our dataset. Out of 195 countries in total, 160 countries seem to have hosted a C2 server at least quite a few times, indicating that threat actors utilize services even from smaller countries. *C2Store* reveals that certain countries have a higher concentration of active C2 addresses, namely the USA (22.7%), China (18.9%), The Netherlands (12.6%), and Russia (11.3%). This trend can be attributed to threat actors hosting C2 servers on popular cloud service providers such as DigitalOcean and AWS, known to have most of their data farms in the USA.

Geographical hotspots: Based on our analysis of multiple malware families, we have observed a concentration of C2 servers in certain geographic locations for specific malware families, which we refer to as "hotspots". Figure 6 highlights a breakdown of the top countries hosting C2 servers across multiple malware families. Due to space limitations, we only show the distribution for 3 families but observed a similar trend for other families. This further emphasizes the hotspot-like nature of different families of C2 servers. This could be due to the fact that threat actors may strategically select certain geographic locations to host their C2 server, possibly due to factors such as proximity to targets or availability of server resources.

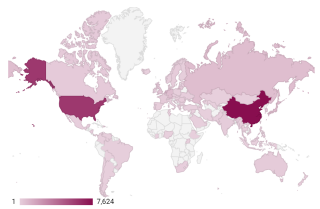
Observation 7*. Several well-known service providers host significant numbers of C2 servers. This suggests that detecting and containing C2 servers is a challenge.*



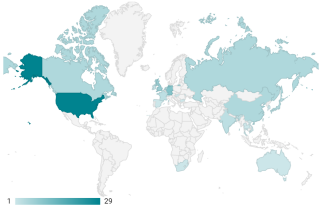
(a) Aurora stealer: Netherlands (33.4%), Russia (23.3%), Germany (20.5%), Finland (5.7%).



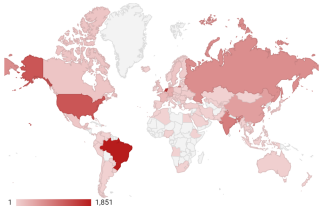
(b) BRC4: Japan (48.5%), USA (22.1%), Hong Kong (11.8%), Germany (4.4%).



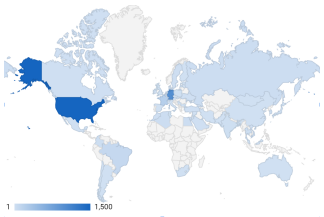
(c) Cobalt Strike: China (32.2%), USA (25.2%), Hong Kong (10.3%), Russia (5.4%).



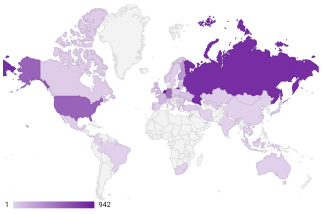
(d) Havoc: USA (28.4%), Germany (10.8%), Singapore (7.8%), UK (7.8%).



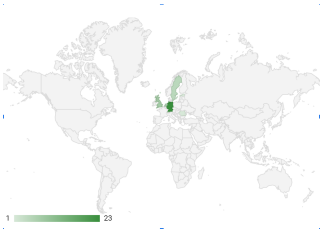
(e) Mirai: Brazil (17.9%), Netherlands (15.5%), USA (12.1%), India (8.3%).



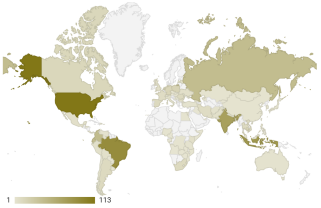
(f) njRAT: USA (46.3%), Germany (31%), France (6.4%), Brazil (2.9%).



(g) Redline Stealer: Netherlands (21.4%), Russia (19.2%), USA (12.9%), Germany (11.6%).



(h) Sharkbot: Germany (45.1%), UK (15.7%), Switzerland (13.7%).



(i) Trickbot: USA (17.6%), Brazil (15%), Indonesia (13.5%), India (12.5%).

Fig. 6. Vastly different geolocations of C2 servers: malware families and their C2 server distributions.

6.1.2 An AS-centric analysis. Based on our analysis of C2 servers, we have observed that there are 3,007 unique ASes hosting C2 servers. However, only a small percentage of these ASes (13.2% or 398 ASes) were found to be hosting more than 500 C2 servers. This suggests that a relatively small number of service providers are being repeatedly utilized for hosting C2 servers. In Table 4, we list the top 10 most active ASes in terms of hosting C2 servers in our dataset.

First, several of the top-10 ASes are well-known service providers that offer hosting services to customers worldwide. This group includes AS 45090 (Shenzhen Tencent), AS 16509 (Amazon), AS 14061 (DigitalOcean), and AS 8075 (Microsoft) in order of decreasing activity. This indicates that threat actors may be utilizing legitimate hosting providers to host their C2 server to blend in and making it more difficult to identify and mitigate the threats.

Second, we find that Delis LLC (AS 211252) is a relatively unknown AS that hosts a significant number of C2 servers. We find no information available about Delis LLC on their website (www.delis.one), which is highly unusual for an AS. In addition, we have noticed that their website does not even have an SSL/TLS certificate installed, which is another red flag. Independent services (such as scamalytics.com) corroborate the "nebulous" reputation of Delis LLC, considering it a

Table 4. The top 10 ASes hosting the largest number of C2 servers across different malware families.

AS Name	ASN	Top Family
Shenzhen Tencent	45090	Cobalt Strike, Meterpreter, Qakbot, PlugX
Amazon	16509	NjRAT, Cobalt Strike, Nanocore RAT, Koadic, Sliver
DigitalOcean	14061	Cobalt Strike, IceId, Meterpreter, Mythic, YerLoader
Hangzhou Alibaba	37963	Cobalt Strike, Sliver, PupyRAT, XtremeRAT
AS-CHOOPA	20473	Cobalt Strike, Qakbot, RedLine Stealer, AsyncRAT
Hetzner Online GmbH	24940	RedLine Stealer, Vidar, Mirai, Cobalt Strike
OVH SAS	16276	Cobalt Strike, RedLine Stealer, Mirai, IcedID, Meterpreter
Delis LLC	211252	Mirai, Cobalt Strike, NanocoreRAT, Remcos
COLORCROSSING	36352	Cobalt Strike, Mirai, Bashlite, AsyncRAT
MICROSOFT	8075	Cobalt Strike, Cerberus, Meterpreter, DarkComet

"potentially high fraud risk AS" with a fraud score of 70%. Clearly, further investigation is needed before a case can be made beyond these initial indications.

AS hotspots. Similar to geographic hotspots (seen in §6.1.1), we have observed that several ASes exhibit hotspot-like behavior. First, of all the C2s in AS 45090, 61% are attributed to Cobalt Strike, which accounts for 20% of all Cobalt Strike C2s. Second, AS 14061 (DigitalOcean) and AS 210269 (HostCircle B.V.) have been identified as a hotspot for the IceID and Mirai botnets, which primarily target Internet of Things (IoT) devices. Third, AS 211409 (Shelter LLC) has 768 IP addresses allocated, out of which 346 were hosting the C2 servers, with 154 (44%) of the Aurora stealer family.

Overall, these findings underscore the importance of AS-level analysis in identifying hotspot ASes for malicious network activity, which can be easily and effectively done using *C2Store*.

Potential extensions. It is important to note that our AS-level analyses do not involve network traffic analysis and rely solely on AS information obtained through industry-standard tools like ipinfo. However, to enhance this aspect, we can consider leveraging real-time probing techniques, as discussed in §3.2. This approach would allow us to directly observe the owner of the IP address in real-time. It could also assist in identifying the duration of malicious activity, particularly in cases where IP addresses are recycled and reassigned to benign users by cloud providers.

6.2 Temporal Analysis

We analyze the temporal properties of C2 servers by answering the following questions:
Q1. What are the temporal trends of the identified C2 servers? (§6.2.1)
Q2. What are the response patterns of C2 servers? (§6.2.2)

Observation 8*. The C2 server reporting exhibits a persistent activity per server family over time.*

6.2.1 Reported C2 servers over time. We study the number of reported C2 servers over time in 2022 (shown in Figure 7). We see that the most prominent families have a roughly consistent level of identified C2 servers per quarter. The Cobalt Strike family shows some decline but remains the dominant family of identified servers. All these families are named after penetration testing tools that are being used to support their proliferation and communication. We find that these tools have been in use in the last 10 years by botnets. This corroborates our earlier observation that botnets rely on old techniques and tools.

Observation 9*. C2 servers are elusive with "unreliable" response patterns.*

6.2.2 The elusive nature of C2 servers. We study the responsiveness of our C2 servers using our probing method, which we explained in §3.2. We conducted an experiment on April 7, 2023, using 1,000 newly reported C2 servers. We then attempt to communicate with these servers, sending six

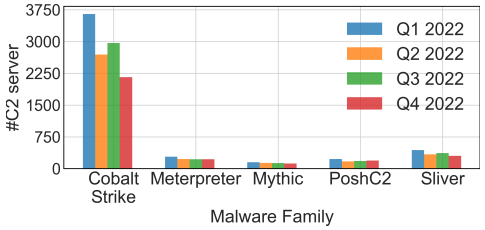


Fig. 7. Number of reported C2 servers over four quarters in 2022 for different families.

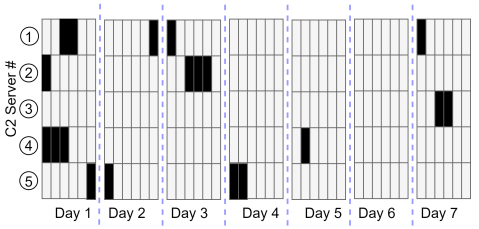


Fig. 8. Elusive C2 Servers: Inconsistent Responses Under Daily Probing.

probes daily, one every four hours over 7 days starting on April 7th 2023. Naturally, we send the appropriate probes for each family type. We find that **95% of these servers responded to fewer than 12% of the probes** (less than 20) over the 7-day period.

Intrigued, we wanted to visualize the server behavior to obtain a more immediate understanding. First, we selected a random representative server from each of the top 5 families: ① Cobalt Strike, ② Qakbot, ③ Mirai, ④ RedLine Stealer and ⑤ njRAT. These families have a substantial presence consistently in our dataset. We then plot the received responses throughout the week in Figure 8. The black boxes represent the cases where the servers responded to our probes. We make two key observations that capture different aspects of the elusive behavior of the servers:

a. The servers do not respond to all the probes within a day. We see that we never see 6 successful probes on any given day. In fact, we don't see more than three successive responses in our probes.

b. The servers do not appear active every day. For example, we see that RedLine Stealer ④ stops responding for 3 days after its responses on the first day but re-engages on day 5.

Practical implications. Our observation could help inform a C2 server monitoring capability and research studies in this direction. Our experiments suggest that the C2 servers do not have a consistent response behavior. A monitoring capability needs to take into consideration this unreliable behavior when determining the frequency of probing. First, a large number of probes needs to be used to assess the presence of a C2 server. Second, any research study should be careful not to rush to assume that the lack of a response guarantees the absence of a server.

6.3 Behavioral Analysis

We use our dataset and capabilities to understand the behavior of C2 servers. Due to space limitations, we limit our discussion to two surprising observations.

6.3.1 ***Multi-family hosting:*** *An IP address hosts C2 servers for more than one family.* We observe an unusual behavior of sharing C2 servers across different malware families. Specifically, we found that 250 IP addresses were running C2 servers for more than 5 different malware families at the same time as of 29th March 2023. We manually verify this information in three ways: (a) by importing different service banners running on different ports and comparing them to well-known signatures, (b) by anonymously visiting the website using a secure VM on a cloud platform and checking the content of the website, and (c) by running Shodan queries [50] (crawled from a Twitter user) and obtaining their banners. These verification processes confirmed this phenomenon. Prominent families in this behavior were Sliver, Cobalt Strike, Mythic, and Deimos, which could indicate a collaborative pattern among the threat actors that use them.

6.3.2 ***Unconventional C2 communication using Pastebin.*** We identified an unusual way for C2 communications. We noticed that 1306 binaries use the well-known Pastebin (pastebin.com) text-sharing platform for C2 communication. We found this by querying our *C2Store* about the

most number of attempted DNS resolutions. We discovered compelling evidence: first, multiple Pastebin URLs were flagged as malicious by AV engines in VT. Second, upon visiting the website, we found evidence of malicious exploits, including SSH brute force and root exploit commands. Note that Pastebin is a benign platform that ranks within the 2000 most popular websites. The use of Pastebin can be seen as an advanced indirect approach to conducting C2 communication, comparable to the use of Internet Relay Chat services by bots. Attackers write their commands on this site, which are then read by their bots. As a result, access to Pastebin is unlikely to be flagged and blocked by firewalls. We have notified the administrators of the Pastebin site.

7 DISCUSSION

We discuss the broader scope and limitations of our approach.

a. Scope/Usage of *C2Store*. We see our *C2Store* as a continuous monitoring system that serves as a key reference point for the security community when looking for C2 servers. It can be used as (a) a source of C2 blacklist information; (b) an automated reporting/alerting tool that can be integrated with existing security tools, and (c) as a source of "fresh" information about C2 server behaviors.

b. Data Poisoning. In our current system, we took a rigorous approach of manually inspecting the list of Twitter users from whom we mined their posted tweets to extract C2 information to mitigate data poisoning concerns. However, more sophisticated and scalable techniques [25, 72, 81, 90] can be applied to detect fake accounts and can be explored in future work.

c. Automation and Real-world deployment. Our current version is deployed and accessible online at https://c2store.github.io/. The majority of services, including tweet and repository fetching and information synthesis, are automated. However, the crucial task of selecting trusted Twitter users and GitHub repositories is performed manually to prevent data poisoning and maintain reliability. To enhance this process, automated techniques can be explored as alluded to in the previous paragraph.

d. Future plans. Our goal is to amplify the ability of security analysts to monitor and study the behavior of C2 servers by providing extensive information in a unified platform. Our plan is to establish *C2Store* as a reference resource by: (a) maintaining a continuous monitoring and alerting system; (b) gathering information from additional sources, such as security forums and a wider range of social media platforms; (c) integrating automated fake account detection; (d) utilizing active scanning methods to locate operational C2 servers; (e) including more metadata, such as type of malicious activity conducted, and (f) soliciting and carefully incorporating feedback from users and the community to improve the quality of our data continuously.

e. Is our dataset representative? This is the question that is asked of any measurement effort. In our case, we propose an initiative as an ongoing effort to continuously collect and synthesize an ever-increasing number of sources. In parallel, we argue that our data is already relatively comprehensive. First, we combine most known and even uncommon sources of information, and as we saw, the union exceeds the coverage of each individual source. Second, the synthesis and cross-validation of the data can provide a level of confidence that the user of the data can select. Our earlier analysis should provide adequate substantiation of the above claims.

f. Can we incorporate more sources of information? Yes. We have designed our approach to be modular and have already seen how integrating information in our Information Synthesis phase is done in a careful way. If the information sources fall into a category that we have already used, the task will require minor adjustments. We already have functions for collecting and processing data, so we will only need to customize our methods to specific features of the new data sources.

g. Limitations and caveats. Although any monitoring capability can always improve, we argue that *C2Store* is already a useful resource for the community regarding the quality and value of the information it provides. As with any data-driven effort, the quality of the information revolves

around: (a) coverage, (b) accuracy, and (c) timeliness. We have already evaluated *C2Store* with respect to all three dimensions, and though not perfect, we argue that it is substantial. Naturally, the quality of *C2Store* will improve as it incorporates more quality sources. We would like to stress again that our social-media-based methods provide substantial and more timely information, which we consider a significant contribution to the work. We intend to continue to look for and incorporate high-quality sources of information. An additional advantage of having multiple and, ideally, independent sources of information is that their collective agreement increases both the coverage and our confidence in the reported C2 servers.

8 RELATED WORKS

There has been limited work addressing the problem of establishing a definitive and continuous C2 server tracking service. Given that this is the focus of our work, we find that most previous efforts and initiatives could have a synergistic relationship with our effort. We discuss prior efforts across the following categories.

a. Threat intelligence feeds: Several Threat feeds [11, 20, 83] offer C2 information; however, these feeds are often limited to a specific set of malware families and are often delayed (as seen in §4.2), limiting their effectiveness. Very few feeds come with limited querying functionality. For example, ThreatFox supports very basic queries, such as filtering according to malware families. Many studies [23, 60] have shown the ineffectiveness of these Threat feeds. In addition, these services do not provide even trivial metadata such as AS and geographic location, which limits the user from getting valuable insights. Moreover, the absence of support for advanced queries impedes the user's ability to obtain detailed insights. In contrast, our *C2Store* provides rich annotations and supports advanced querying capabilities, enabling users to obtain more detailed and comprehensive insights into C2 servers.

b. Binary-enabled search for C2 servers: Although some efforts have been made to find live C2 servers using malware binaries [29, 30] or tools like Shodan [64, 78], the search process can be challenging without a starting point or clue. Our solution, *C2Store*, can assist in this approach by providing guidance on potential IP ranges to scan when searching for live C2 servers. By utilizing hotspot information, our solution can identify regions and IP addresses, allowing for the more targeted and efficient active scanning of specific families, ultimately saving time.

Note that many studies have analyzed various aspects of malware binaries and, in the process, identified C2 servers but not with the intention of collecting and building a reference C2 capability. Some efforts study specific families such as Mirai [13, 40] and Hajime [44], while other efforts study aspects of the malware lifecycle and its phases [10, 12, 26, 35, 43, 58, 86].

c. Mining social media and software archives. Most previous efforts in this category have focused on extracting different types of security information, but not C2 servers. In more detail, there have been studies that mined social media platforms like Twitter, but they focus on identifying malicious actors [46, 47], characterizing their ecosystem [38, 59, 69, 82], or detecting the emergence of threats [37, 47, 75]. In addition, other studies have analyzed software archives like GitHub to identify malicious source code, binaries, [74], but not C2 servers as we do here.

d. Traffic-based C2 server detection: There have been several studies [31, 41, 42, 48, 57] that aim to detect C2 traffic by analyzing packet captures on networks. These studies typically use machine learning or other data analysis techniques to identify traffic patterns characteristic of C2 communication. In addition to this, some efforts try to detect different various beacons generated by red-teaming tools such as Cobalt strike [85], Meterpreter [84]. However, while these studies have yielded promising results, there has been limited focus on analyzing and characterizing the C2 servers in the wild, which we intend to do with our work.

We see our work as synergistic with the above prior efforts: (a) we can incorporate their findings in our database, and (b) we can provide them with information to improve the effectiveness of their techniques by providing with archival and spatiotemporal information on C2 servers.

9 CONCLUSION

The goal of our work is to develop an open-access capability to provide the most comprehensive information on C2 servers. Containing botnets will require multiple technical and policy efforts, and we argue that tracking C2 servers is an essential capability. The contributions of our work are threefold: (a) we develop techniques to collect, sanitize, and combine C2 server addresses from five types of sources; (b) we create a continuously updating database with 335,967 C2 servers across 133 malware families; and (c) we showcase the usefulness of our capability by identifying interesting behaviors of C2 servers. Due to space limitations, we omit a recap of the key results that are already listed in the introduction.

Our ambitious goal is to establish *C2Store* (https://c2store.github.io/) as the de-facto C2 server reference resource for the security community by continuously updating its scope and reach.

ACKNOWLEDGMENTS

We thank our anonymous shepherd and reviewers for their valuable suggestions and comments. This research was partly supported by NSF SaTC Grant No. 2132642. This research was also sponsored partly by the OUSD(R&E)/RT&L and was accomplished under Cooperative Agreement Number W911NF-20-2-0267. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ARL and OUSD(R&E)/RT&L or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein.

REFERENCES

[1] 2023. Feodo Tracker. https://feodotracker.abuse.ch/. (Accessed on 05/01/2023).
[2] 2023. GeoIP® Databases & Services: Industry Leading IP Intelligence | MaxMind. https://www.maxmind.com/en/geoip2-services-and-databases. (Accessed on 05/01/2023).
[3] 2023. GitHub. https://github.com/. (Accessed on 05/01/2023).
[4] 2023. Introduction — PyGithub. https://pygithub.readthedocs.io/en/latest/introduction.html. (Accessed on 05/20/2023).
[5] 2023. tesseract-ocr/tesseract: Tesseract Open Source OCR Engine (main repository). https://github.com/tesseract-ocr/tesseract. (Accessed on 05/10/2023).
[6] 2023. The trusted source for IP address data, leading IP data provider - IPinfo.io. https://ipinfo.io/. (Accessed on 05/02/2023).
[7] 2023. twarc. https://twarc-project.readthedocs.io/en/latest/. (Accessed on 05/08/2023).
[8] 2023. VirusTotal. https://www.virustotal.com/. (Accessed on 05/01/2023).
[9] 2023. www.blocklist.de – Fail2Ban-Reporting Service. https://www.blocklist.de/en/index.html. (Accessed on 05/03/2023).
[10] Arwa Abdulkarim Al Alsadi, Kaichi Sameshima, Jakob Bleier, Katsunari Yoshioka, Martina Lindorfer, Michel Van Eeten, and Carlos Hernandez Gañán. 2022. No Spring Chicken: Quantifying the Lifespan of Exploits in IoT Malware Using Static and Dynamic Analysis.. In *AsiaCCS*. 309–321.
[11] Alienvault. 2023. Alienvault - Open Threat Exchange. https://otx.alienvault.com/
[12] Omar Alrawi, Charles Lever, Kevin Valakuzhy, Ryan Court, Kevin Z Snow, Fabian Monrose, and Manos Antonakakis. 2021. The Circle Of Life: A Large-Scale Study of The IoT Malware Lifecycle.. In *USENIX Security Symposium*. 3505–3522.
[13] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. 2017. Understanding the mirai botnet. In *26th {USENIX} security symposium ({USENIX} Security 17)*. 1093–1110.
[14] Hadi Asghari. 2020. pyasn. https://catalog.caida.org/software/pyasn. Accessed: 2023-5-2.
[15] Guest Author. 2017. Inside the infamous Mirai IoT Botnet: A Retrospective Analysis. https://blog.cloudflare.com/inside-mirai-the-infamous-iot-botnet-a-retrospective-analysis/. (Accessed on 06/25/2023).

[16] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. 2012. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*. 519–528.
[17] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. 2009. Gephi: An Open Source Software for Exploring and Manipulating Networks. http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154
[18] Benkow. 2023. Benkow. https://benkow.cc/index.php. (Accessed on 05/03/2023).
[19] Leyla Bilge, Davide Balzarotti, William Robertson, Engin Kirda, and Christopher Kruegel. 2012. Disclosure: detecting botnet command and control servers through large-scale netflow analysis. In *Proceedings of the 28th Annual Computer Security Applications Conference*. 129–138.
[20] ViriBack Blog. 2023. Malware tracker, Iocs & More. https://viriback.com/
[21] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.
[22] Harold Booth, Doug Rike, and Gregory A Witte. 2013. The national vulnerability database (nvd): Overview. (2013).
[23] Xander Bouwman, Harm Griffioen, Jelle Egbers, Christian Doerr, Bram Klievink, and Michel Van Eeten. 2020. A different cup of TI? The added value of commercial threat intelligence. In *Proceedings of the 29th USENIX Conference on Security Symposium*. 433–450.
[24] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems* 30, 1-7 (1998), 107–117.
[25] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. 2016. Debot: Twitter bot detection via warped correlation.. In *Icdm*, Vol. 18. IEEE, 28–65.
[26] Jinchun Choi, Afsah Anwar, Hisham Alasmary, Jeffrey Spaulding, DaeHun Nyang, and Aziz Mohaisen. 2019. Iot malware ecosystem in the wild: a glimpse into analysis and exposures. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. 413–418.
[27] Valerio Cosentino, Javier Luis, and Jordi Cabot. 2016. Findings from GitHub: methods, datasets and limitations. In *Proceedings of the 13th International Conference on Mining Software Repositories*. 137–141.
[28] Ahmad Darki and Michalis Faloutsos. 2020. RIoTMAN: a systematic analysis of IoT malware behavior. In *International Conference On Emerging Networking Experiments And Technologies (CoNEXT)*. ACM, 169–182.
[29] Ali Davanian, Ahmad Darki, and Michalis Faloutsos. 2021. CnCHunter: An MITM-approach to identify live CnC servers. *BlackHat USA* (2021).
[30] Ali Davanian and Michalis Faloutsos. 2022. MalNet: A binary-centric network-level profiling of IoT malware. In *Proceedings of the 22nd ACM Internet Measurement Conference*. 472–487.
[31] Lorenzo De Carli, Ruben Torres, Gaspar Modelo-Howard, Alok Tongaonkar, and Somesh Jha. 2017. Botnet protocol inference in the presence of encrypted traffic. In *IEEE INFOCOM 2017 Conference on Computer Communications*. IEEE, 1–9.
[32] Malware Must Die! 2016. MMD-0056-2016 - Linux/Mirai, how an old ELF malcode is recycled.. https://blog.malwaremustdie.org/2016/08/mmd-0056-2016-linuxmirai-just.html. (Accessed on 06/25/2023).
[33] drb ra. 2023. C2IntelFeeds: Automatically created C2 Feeds. https://github.com/drb-ra/C2IntelFeeds. (Accessed on 06/27/2023).
[34] Zakir Durumeric, David Adrian, Ariana Mirian, Michael Bailey, and J Alex Halderman. 2015. A search engine backed by Internet-wide scanning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 542–553.
[35] Aaron Faulkenberry, Athanasios Avgetidis, Zane Ma, Omar Alrawi, Charles Lever, Panagiotis Kintis, Fabian Monrose, Angelos D Keromytis, and Manos Antonakakis. 2022. View from Above: Exploring the Malware Ecosystem from the Upper DNS Hierarchy. In *Proceedings of the 38th Annual Computer Security Applications Conference*. ACM, 240–250.
[36] Jerry Gamblin. 2017. jgamblin/Mirai-Source-Code: Leaked Mirai Source Code for Research/IoC Development Purposes. https://github.com/jgamblin/Mirai-Source-Code. (Accessed on 05/24/2023).
[37] Joobin Gharibshah, Tai-Ching Li, Maria Solanas Vanrell, Andre Castro, Konstantinos Pelechrinis, Evangelos E. Papalexakis, and Michalis Faloutsos. 2017. InferIP: Extracting actionable information from security discussion forums. In *IEEE/ACM ASONAM*.
[38] Joobin Gharibshah, Evangelos Papalexakis, and Michalis Faloutsos. 2020. REST: A thread embedding approach for identifying and classifying user-specified information in security forums. *ICWSM* (2020).
[39] Robert David Graham. 2023. TCP port scanner, spews SYN packets asynchronously, scanning entire Internet in under 5 minutes. https://github.com/robertdavidgraham/masscan. (Accessed on 05/19/2023).
[40] Harm Griffioen and Christian Doerr. 2020. Examining mirai's battle over the internet of things. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 743–756.
[41] Guofei Gu, Vinod Yegneswaran, Phillip Porras, Jennifer Stoll, and Wenke Lee. 2009. Active botnet probing to identify obscure command and control channels. In *2009 annual computer security applications conference*. IEEE, 241–253.

[42] Guofei Gu, Junjie Zhang, and Wenke Lee. 2008. BotSniffer: Detecting botnet command and control channels in network traffic. (2008).

[43] Huy Hang, Xuetao Wei, Michalis Faloutsos, and Tina Eliassi-Rad. 2013. Entelecheia: Detecting p2p botnets in their waiting stage. In *2013 IFIP Networking Conference*. IEEE, 1–9.

[44] Stephen Herwig, Katura Harvey, George Hughey, Richard Roberts, and Dave Levin. 2019. Measurement and analysis of Hajime, a peer-to-peer IoT botnet. In *Network and Distributed Systems Security (NDSS) Symposium*.

[45] Sameera Horawalavithana, Abhishek Bhattacharjee, Renhao Liu, Nazim Choudhury, Lawrence O. Hall, and Adriana Iamnitchi. 2019. Mentions of security vulnerabilities on reddit, twitter and github. In *IEEE/WIC/ACM International Conference on Web Intelligence*. 200–207.

[46] Risul Islam, Md Omar Faruk Rokon, Ahmad Darki, and Michalis Faloutsos. 2020. HackerScope: The Dynamics of a Massive Hacker Online Ecosystem. In *Proceedings of International Conference on Advances in Social Network Analysis and Mining (ASONAM)*. IEEE/ACM.

[47] Risul Islam, Md Omar Faruk Rokon, Evangelos E. Papalexakis, and Michalis Faloutsos. 2020. TenFor: A Tensor-Based Tool to Extract Interesting Events from Security Forums. In *Proceedings of International Conference on Advances in Social Network Analysis and Mining (ASONAM - industrial track)*. IEEE/ACM.

[48] Gregoire Jacob, Ralf Hund, Christopher Kruegel, and Thorsten Holz. 2011. JACKSTRAWS: Picking Command and Control Connections from Bot Traffic.. In *USENIX Security Symposium*, Vol. 2011. San Francisco, CA, USA.

[49] Octave Klaba. 2016. Octave Klaba on Twitter. https://twitter.com/olesovhcom/status/778830571677978624. (Accessed on 06/25/2023).

[50] Michael Koczwara. 2023. Hunting C2. Hunting C2/Adversaries Infrastructure | Medium. https://michaelkoczwara.medium.com/hunting-c2-with-shodan-223ca250d06f. (Accessed on 05/01/2023).

[51] John Kurkowski. 2023. john-kurkowski/tldextract: Accurately separates a URL's subdomain, domain, and public suffix, using the Public Suffix List (PSL). https://github.com/john-kurkowski/tldextract. (Accessed on 05/01/2023).

[52] Grafana Labs. 2023. Grafana: The open observability platform. https://grafana.com/. (Accessed on 06/26/2023).

[53] Renaud Lambiotte, J-C Delvenne, and Mauricio Barahona. 2008. Laplacian dynamics and multiscale modular structure in networks. *arXiv preprint arXiv:0812.1770* (2008).

[54] Victor Le Pochat, Sourena Maroofi, Tom Van Goethem, Davy Preuveneers, Andrzej Duda, Wouter Joosen, Maciej Korczyński, et al. 2020. A practical approach for taking down avalanche botnets under real-world constraints. In *Proceedings of the 27th Annual Network and Distributed System Security Symposium*. Internet Society.

[55] Kristina Lerman and Rumi Ghosh. 2010. Information contagion: An empirical study of the spread of news on digg and twitter social networks. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 4. 90–97.

[56] Edouard Leurent. 2023. eleurent/twitter-graph: Fetch and visualize the graph of your Twitter friends and followers. https://github.com/eleurent/twitter-graph. (Accessed on 05/01/2023).

[57] Chaz Lever, Platon Kotzias, Davide Balzarotti, Juan Caballero, and Manos Antonakakis. 2017. A lustrum of malware network communication: Evolution and insights. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 788–804.

[58] Hongda Li, Qiqing Huang, Fei Ding, Hongxin Hu, Long Cheng, Guofei Gu, and Ziming Zhao. 2022. Understanding and Detecting Remote Infection on Linux-based IoT Devices. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. 873–887.

[59] Tai-Ching Li, Joobin Gharibshah, Evangelos E. Papalexakis, and Michalis Faloutsos. 2017. TrollSpot: Detecting misbehavior in commenting platforms. In *IEEE/ACM ASONAM*.

[60] Vector Guo Li, Matthew Dunn, Paul Pearce, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. 2019. Reading the tea leaves: A comparative analysis of threat intelligence. In *28th USENIX security symposium (USENIX Security 19)*. 851–867.

[61] Martina Lindorfer, Clemens Kolbitsch, and Paolo Milani Comparetti. 2011. Detecting environment-sensitive malware. In *Recent Advances in Intrusion Detection: 14th International Symposium, RAID 2011*. Springer, 338–357.

[62] MalwareBazaar. 2023. MalwareBazaar | Malware sample exchange. https://bazaar.abuse.ch/. (Accessed on 05/01/2023).

[63] László Molnár Markus F.X.J. Oberhumer and John F. Reiser. 2023. UPX: the Ultimate Packer for eXecutables - Homepage. https://upx.github.io/. (Accessed on 05/01/2023).

[64] John Matherly. 2015. Complete guide to shodan. *Shodan, LLC (2016-02-25)* 1 (2015).

[65] Aziz Mohaisen, Omar Alrawi, and Manar Mohaisen. 2015. AMAL: high-fidelity, behavior-based automated malware analysis and classification. *computers & security* 52 (2015), 251–266.

[66] Yacin Nadji, Manos Antonakakis, Roberto Perdisci, David Dagon, and Wenke Lee. 2013. Beheading hydras: performing effective botnet takedowns. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*.

[67] Krebs on Security. 2016. KrebsOnSecurity Hit With Record DDoS. https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/. (Accessed on 06/25/2023).

[68] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2018. Tranco: A research-oriented top sites ranking hardened against manipulation. *arXiv preprint arXiv:1806.01156* (2018).

[69] Rebecca S Portnoff, Sadia Afroz, Greg Durrett, Jonathan K Kummerfeld, Taylor Berg-Kirkpatrick, Damon McCoy, Kirill Levchenko, and Vern Paxson. 2017. Tools for automated analysis of cybercriminal markets. In *Proceedings of the 26th international conference on world wide web*. 657–666.
[70] YaraRules Project. 2021. Rules to catch Malware. https://yara-rules.github.io/blog/. (Accessed on 06/27/2023).
[71] Kevin Pulo. 2009. Fun with LD_PRELOAD. In *linux. conf. au*, Vol. 153. 103.
[72] Anirudh Ramachandran, Nick Feamster, and Santosh Vempala. 2007. Filtering spam with behavioral blacklisting. In *Proceedings of the 14th ACM conference on computer and communications security*. 342–351.
[73] Konrad Rieck, Thorsten Holz, Carsten Willems, Patrick Düssel, and Pavel Laskov. 2008. Learning and classification of malware behavior. In *Detection of Intrusions and Malware, and Vulnerability Assessment: 5th International Conference, DIMVA 2008, Paris, France, July 10-11, 2008. Proceedings 5*. Springer, 108–125.
[74] Md Omar Faruk Rokon, Risul Islam, Ahmad Darki, Evangelos E Papalexakis, and Michalis Faloutsos. 2020. SourceFinder: Finding Malware Source-Code from Publicly Available Repositories in GitHub.. In *RAID*. 149–163.
[75] Parang Saraf and Naren Ramakrishnan. 2016. EMBERS autogsr: Automated coding of civil unrest events. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 599–608.
[76] Marcos Sebastián, Richard Rivera, Platon Kotzias, and Juan Caballero. 2016. Avclass: A tool for massive malware labeling. In *Research in Attacks, Intrusions, and Defenses: 19th International Symposium, RAID 2016, Paris, France, September 19-21, 2016, Proceedings 19*. Springer, 230–253.
[77] Silvia Sebastián and Juan Caballero. 2020. Avclass2: Massive malware tag extraction from av labels. In *Annual Computer Security Applications Conference*. 42–53.
[78] Shodan. 2023. Shodan Search Engine. https://www.shodan.io/. (Accessed on 05/02/2023).
[79] Ray Smith et al. 2007. Tesseract ocr engine. *Lecture. Google Code. Google Inc* (2007).
[80] Stratosphere. 2015. Stratosphere Laboratory Datasets. Retrieved March 13, 2020, from https://www.stratosphereips.org/datasets-overview.
[81] Venkatramanan S Subrahmanian, Amos Azaria, Skylar Durst, Vadim Kagan, Aram Galstyan, Kristina Lerman, Linhong Zhu, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2016. The DARPA Twitter bot challenge. *Computer* 49, 6 (2016), 38–46.
[82] Jakapun Tachaiya, Arman Irani, Kevin M. Esterling, and Michalis Faloutsos. 2021. SentiStance: quantifying the intertwined changes of sentiment and stance in response to an event in online forums. In *ASONAM '21: International Conference on Advances in Social Networks Analysis and Mining, Virtual Event, The Netherlands, November 8 - 11, 2021*, Michele Coscia, Alfredo Cuzzocrea, Kai Shu, Ralf Klamma, Sharyn O'Halloran, and Jon G. Rokne (Eds.). ACM, 361–368. https://doi.org/10.1145/3487351.3490966
[83] ThreatFox. 2023. ThreatFox: Share indicators of compromise (iocs). https://threatfox.abuse.ch/ (Accessed on 05/01/2023).
[84] Nicholas Troutman. 2018. *Detecting Meterpreter Traffic*. Technical Report. Sandia National Lab.(SNL-NM), Albuquerque, NM (United States).
[85] Vincent van der Eijk and Coen Schuijt. 2019. Detecting cobalt strike beacons in NetFlow data. (2019).
[86] Pierre-Antoine Vervier and Yun Shen. 2018. Before toasters rise up: A view into the emerging IoT threat landscape. In *Research in Attacks, Intrusions, and Defenses: 21st International Symposium, RAID 2018, Heraklion, Crete, Greece, September 10-12, 2018, Proceedings 21*. Springer, 556–576.
[87] VirusShare. 2023. VirusShare.com. https://virusshare.com/. (Accessed on 05/01/2023).
[88] VirusTotal. 2023. VirusTotal - Top users. https://www.virustotal.com/gui/top-users. (Accessed on 05/02/2023).
[89] Common Vulnerabilities and Exposures. 2023. CVE. https://cve.mitre.org/index.html. (Accessed on 06/21/2023).
[90] Xichen Zhang and Ali A Ghorbani. 2020. An overview of online fake news: Characterization, detection, and discussion. *Information Processing & Management* 57, 2 (2020), 102025.
[91] Shuofei Zhu, Jianjun Shi, Limin Yang, Boqin Qin, Ziyi Zhang, Linhai Song, and Gang Wang. 2020. Measuring and Modeling the Label Dynamics of Online Anti-Malware Engines.. In *USENIX Security Symposium*. 2361–2378.

Received July 2023; accepted October 2023