IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 3 February 2023, accepted 17 February 2023, date of publication 27 February 2023, date of current version 13 March 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3249474*

RESEARCH ARTICLE

# Detecting Coordinated Internet-Wide Scanning by TCP/IP Header Fingerprint

**AKIRA TANAKA, CHANSU HAN, AND TAKESHI TAKAHASHI, (Member, IEEE)**
National Institute of Information and Communications Technology, Koganei 184-8795, Japan

Corresponding author: Akira Tanaka (tanaka.akira@nict.go.jp)

This work was supported by the Ministry of Internal Affairs and Communications, Japan, under a Contract of "MITIGATE" among "Research and Development for Expansion of Radio Wave Resources" under Grant JPJ000254.

**ABSTRACT** Adversaries perform port scanning to discover accessible and vulnerable hosts as a prelude to cyber havoc. A darknet is a cyberattack observation network to capture these scanning activities through reachable yet unused IP addresses. However, the enormous amount of packets and superposition of diverse scanning strategies prevent extracting significant insights from the aggregate traffic. Some coordinated scanners disperse probe packets whose TCP/IP header follows a unique pattern to determine whether the received packets are valid responses to their probes or are part of other background traffic. We call such a pattern a fingerprint. For example, a probe packet from a Mirai-infected host satisfies a pattern whereby the destination IP address equals the sequence number. A fingerprint indicates that the source host has been involved in a particular scanning campaign. Although some fingerprints have been discovered and known to the public, there are and will be more undiscovered ones. We intend to unveil these fingerprints. Our preliminary work automatically identified flexible fingerprints but overlooked low-rate and coordinated scanners. In this work, we improved the fingerprint identifier, enabling it to detect these stealth scans. Moreover, we revealed the scans' objectives by investigating destination port sets. We associated fingerprints with threat intelligence and verified their reliability. Our approach identified all well-known and eight unknown fingerprints on one month's worth of darknet data collected from about three-hundred thousand unused IP addresses. We disclosed the fingerprints of the Mozi botnet and destination port sets that were previously unreported.

**INDEX TERMS** Clustering, darknet (network telescope), fingerprint, port set analysis, scanning campaigns.

## I. INTRODUCTION

Because of the growing use of insecure Internet of Things (IoT) devices [1], [2], IoT-tailored malware and botnets have recently increased in number and activity. For example, in 2018, Torabi et al. [1] exposed 26,000 compromised IoT devices, of which 40% were active in critical infrastructure. Malware and botnets exploit vulnerable IoT devices and perform further malicious activities such as spamming, phishing, cryptojacking, stealing private information, and distributed denial of service (DDoS) attacks. The Mirai botnet [3] has controlled hundreds of thousands of IoT devices to induce a spree of massive high-profile DDoS attacks since late 2016. Surprisingly, the initial attack on Krebs [4] exceeded 600 GBps in volume. Mirai has spawned many variants that follow the same infection strategy. Echobot [5] is a variant whose campaign has infected devices across more than ten vendors by exploiting more than 20 unique (software and firmware) IoT-centric vulnerabilities. Such IoT-tailored malware propagates by scanning the Internet for vulnerable and exploitable IoT devices to recruit them into coordinated IoT botnets. To mitigate and prevent large-scale cyber attacks, we need to identify compromised devices that search for vulnerable devices. One effective approach is observing Internet traffic with passive sensors that collect one-way traffic reaching unused IP addresses.

The darknet [6], [7], a network telescope, passively monitors network traffic with an unreachable dark IP

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

address block. Because of its nature, the darknet receives unsolicited packets from (a) botnets looking for accessible and vulnerable hosts [3], [8], (b) security companies and researchers that build maps of the IPv4 Internet (Shodan[1] and Censys[2]), (c) misconfigured hosts caused by hardware/software errors or improper routings, and (d) the Internet backscatter resulting from DDoS attacks using multiple spoofed addresses [9], [10]. The darknet is an effective system for observing indiscriminate Internet-wide scans and preparing for emerging cyberattacks. However, the massive amount of packets and the superposition of the diverse patterns generated by each scan group prevents meaningful insights from being extracted. Therefore many studies have aimed to specify malicious and survey-objective scanner groups from aggregate darknet traffic.

Generally, orchestrated scanners are supposed to have the following scanning behaviors.

- Scanners target the same destination ports.
- Scanners demonstrate a spatial-temporal correlated scanning pattern [11], [12].
- The packets from cooperating scanners have the same fingerprint, i.e., a unique pattern appearing in the packet header fields (see next paragraph.)

To scrutinize a scanner's behavior, almost all researchers have focused on the scanning features such as (1) destination ports, (2) scan rate, (3) header fields, (4) synchronization, and (5) fingerprints. These features are key indicators of well-coordinated scanner groups. To quantify the features, the early research utilized simple statistics, whereas more recent studies have applied machine-learning and deep-learning techniques. Table 1 summarizes the techniques and input features used by the recent studies that highlight correlated scanning hosts. These studies have discovered many distinctive and coordinated scanner groups, but have had several shortcomings. For instance, the approaches using the scanner's features, except for (5) fingerprints, trigger false positives, which means that unrelated scanning hosts are falsely regarded as orchestrated scanners. The next paragraph details the previous studies on (5) fingerprints; the research on other features is elaborated in Section II.

Scanners often determine whether the received packets are valid responses to their probes or are part of other background traffic. Hence, some scanners encode a unique pattern into each probe packet's TCP/IP header for low-cost re-identification. We call such a pattern a fingerprint. For example, a scanner infected by the Mirai botnet sends probe packets where a bitwise XOR between the sequence number and destination IP address equals zero. Because a fingerprint is a unique pattern for each scan tool, it is more reliable than other features. Previous research [3], [41] created fingerprints based on artifacts of open-source codes. However, some tools might be unknown and unavailable to the general public; for instance, some may have been custom-developed by advanced adversaries to avoid such signature-based detection. Griffioen and Doerr [43] proposed a method for finding a fingerprint, but they assume a fixed-form fingerprint, thereby overlooking sophisticated malware scans with diverse fingerprints. Shaikh et al. [42] applied the context triggered piecewise hashing (CTPH) algorithm [33] to the IP header in order to obtain the signatures of compromised IoT devices. The algorithm divides the file into multiple segments and computes hashes for each segment. Hence, localized segment changes do not affect the hashes for the rest of the file. The authors finally generated IoT-specific empirical attack signatures. Although the CTPH algorithm is more flexible than a fixed-form fingerprint, it cannot handle operations on header fields such as the bitwise XOR. Lastovicka et al. [45] proposed operating system (OS) fingerprinting methods using TCP/IP headers in real networks, to which users can connect any device. They only utilized fields that depend solely on the OS kernel, initial SYN packet size (synSize), window size (winSize), and time to live (TTL). The method calculates the appearance ratio of the triple (synSize, winSize, TTL) for each OS and regards the triple with the highest appearance ratio as the fingerprint of the OS. The method has high coverage but low accuracy. Their fingerprint is a specific tuple form (synSize, winSize, TTL) and hence lacks flexibility in the sense that it does not take another format, such as the bitwise XOR between some fields.

Our preliminary work [46] devised a method for identifying fingerprints that are represented by TCP/IP headers and operations on them. Although the method identified most of the well-known and some unknown fingerprints, it overlooked low-rate and coordinated scanners. Hence, we undertook to improve the method so that it could detect these scanners. Moreover, we revealed the scans' intentions by investigating the destination port sets. Most fingerprints are associated with trustworthy and reputable threat intelligence to verify their reliability. In summary, our major contributions are as follows.

- We identified all well-known fingerprints (Mirai [3], Hajime [47], Masscan [48], and ZMap [49]) and eight unknown fingerprints that included low-rate and coordinated scanners. In particular, we discovered the fingerprints of the Mozi botnet and destination port sets, both of which were previously unreported.
- Scanners with identical fingerprint were distributed in various networks.
- Most fingerprints had unique destination port sets.
- Some scanners had various fingerprints, and we could distinguish distinct campaigns from such a scanner.
- We identified IP addresses that were infected by several botnets simultaneously.

## II. RELATED WORK

Darknet analysis has been discussed in the literature from different perspectives; this section reviews the literature finding cooperating senders that use the scanning features

[1] https://www.shodan.io/
[2] https://censys.io/

**TABLE 1. Techniques and input features used by recent studies that identify well-coordinated scanners. The "dport" column in the scanner's features indicates the destination port. "DBSCAN" in the technique column stands for density-based spatial clustering of applications with noise [13].**

| Approach | Paper | Scanner's features | | | | | Techniques |
|---|---|---|---|---|---|---|---|
| | | dport | #Packet/ Scan rate | Header fields | Synchro-nization | Finger-print | |
| Port-based | Ban et al. [14] | ✓ | | | | | Association rule learning [15] |
| | Lagraa et al. [16] | ✓ | | | | | Modularity clustering [17], Association rule learning [15] |
| | Soro et al. [18] | ✓ | | | | | Modularity clustering [17] |
| | Cohen et al. [19] | ✓ | | | | | Word2Vec [20], DBSCAN [13] |
| | Gioacchini et al. [21] | ✓ | | | | | Word2Vec [20] |
| | Ring et al. [22] | ✓ | | ✓ | | | Word2Vec [20] |
| Time-series | Bou-Harb et al. [23] | | ✓ | | ✓ | | Detrended fluctuation analysis (DFA) [24] |
| | Han et al. [25] | | ✓ | | ✓ | | Graphical lasso [26] |
| | Han et al. [27] | | ✓ | | ✓ | | Non-negative matrix factorization (NMF) [28] |
| | Kanehara et al. [29] | ✓ | ✓ | | ✓ | | Nonnegative Tucker decomposition (NTD) [30] |
| | Fachkha et al. [31] | ✓ | ✓ | ✓ | ✓ | | Dynamic time warping (DTW) [32] and context triggered piecewise hashing (CTPH) [33] |
| Behavior -based | Gates [34] | ✓ | | ✓ | ✓ | | A variant of set covering problem |
| | Bou-Harb et al. [35] | | | ✓ | | | Discrete Fourier transform [36] and Kalman filter [37] |
| | Torabi et al. [38] | ✓ | ✓ | ✓ | | | DBSCAN [13] |
| | Safaei Pour et al. [39] | | ✓ | ✓ | | | Probabilistic model, threshold random walk [40], convolutional neural network (CNN), random forest (RF), and hierarchical agglomerative clustering |
| Fingerprint -based | Durumeric et al. [41] | | | | | ✓ | Inspect open-source codes |
| | Antonakakis et al. [3] | | | | | ✓ | Inspect open-source codes |
| | Shaikh et al. [42] | | | ✓ | | ✓ | Context triggered piecewise hashing (CTPH) [33] |
| | Griffioen and Doerr [43] | | ✓ | ✓ | ✓ | ✓ | Speaker-listener label propagation algorithm (SLPA) [44] |

summarized in Table 1. We classify the related work into port-based, time-series, behavior-based, and fingerprint-based approaches. Because the fingerprint-based approach is dealt with in the previous section, the succeeding subsections describe the research on the three approaches but first let us briefly overview them. The port-based approaches suppose that the destination ports provide deterministic information that can be used to identify unsolicited co-related scanning hosts because different malware scan different combinations of vulnerable ports. Therefore, they perform frequent pattern mining or embedding on destination ports. The time-series approaches endeavor to detect frequent spatiotemporal patterns and synchronization from the packet transmission strategies of scanners. The behavior-based approaches focus on scanning characteristics calculated from the TCP/IP header fields to reveal well-coordinated botnets with similar scanning objectives.

### A. PORT-BASED APPROACHES

Ban et al. [14] applied association rule learning [15] to darknet traffic and explored the correlation among destination ports. The discovered association rules represented regularities among the scanning behaviors of known botnet attacks. Lagraa et al. [16] proposed an unsupervised graph-based model to track port scanning behavior patterns among multiple proved ports. The model performs modularity-based clustering [17] and identifies vertical and horizontal port scanning. The authors enriched a found cluster with metadata (such as geolocation, organization, and domain) and discovered frequently occurring patterns by using association rule learning. Soro et al. [18] also applied modularity-based clustering, but unlike Lagraa's study [16], their graph-based approach was simple. They represented the darknet traffic as a bipartite graph linking traffic sources to the contacted destination ports. Their framework disclosed coordinated IP addresses that predominantly targeted specific ports. They discussed the composition, characteristics, and peculiarities of the found source group.

Recent three studies [19], [21], [22] have utilized an embedding technique, Word2Vec [20], to identify source IP addresses that are coordinated and that engaged in similar activities, such as malicious port scans. Given many sentences, each of which is a sequence of words, Word2Vec returns a vector representation for each word such that the distance between two vectors is small if the two corresponding words appear near each other in many sentences. In the context of darknet analysis, the destination port or source IP address can be regarded as a word, and the above three papers defined words and sentences as summarized in Table 2. In IP2Vec [22], words are defined as unions of source and destination IP addresses, destination ports, and protocols, whereas a sentence is a sequence consisting of these four fields. IP2Vec gets vectors corresponding to the value of each field. This approach was shown to be able to capture the similarity of source IP addresses from botnet data, but a scalability problem remained due to the enormous number of words. DANTE [19] obtains a vector representation of

**TABLE 2.** ***Words*** **and** ***sentences*** **for each algorithm.**

| Algorithm | Word | Sentence |
|---|---|---|
| IP2Vec [22] | ip.src, ip.dst, dport, protocol | the sequence of (ip.src, ip.dst, dport, protocol) of a packet |
| DANTE [19] | dport | a sequence of port numbers of packets sent by an IP address |
| DarkVec [21] | ip.src | a sequence of IP addresses that aim at the same service |

destination ports from sequences of ports sent by an IP address. Then, it defines the vector of an IP address as the average vector of the destination port's vectors. The authors applied DBSCAN [13], a clustering algorithm, to the vectors of IP addresses. DarkVec [21] defines a sentence as a sequence of IP addresses that are aimed at the same service, where the destination port determines the service. It embeds an IP address only once and, unlike DANTE, does not take the average of the embedded vectors, which reduces computation costs and information loss. The authors demonstrated that DarkVec outperformed both DANTE and IP2Vec.

### B. TIME-SERIES APPROACHES

Bou-Harb et al. [23] used the detrended fluctuation analysis (DFA) technique [24] to probe packets from a sender. DFA detects intrinsic self-similarity embedded in a nonstationary time series. The authors identified scanning tools and observed some statistical features (monotonicity and randomness) of probe packets from probing tools, worms, and botnets; however, their approach does not distinguish between worms and botnets. Fachkha et al. [31] devised a novel probabilistic model to sanitize darknet data. The model filters out misconfiguration traffic and has been used in other studies. The authors performed a time-series analysis involving dynamic time warping (DTW) [32] and context triggered piecewise hashing (CTPH) [33] to infer, characterize, and cluster orchestrated and well-coordinated probing activities. They uncovered nearly nine thousand large-scale stealthy, previously undocumented orchestrated proving events targeting many cyber-physical systems (CPS). Han et al. [25] estimated the synchronization degree of packet transmissions among hosts to capture cooperating scanning campaigns in real time. The authors leveraged a graph-based approach called graphical lasso [26] to measure the synchronization and to detect anomalies. In their experiments, their approach detected cyber threats with an accuracy of 97.14%. Another research [27] utilized non-negative matrix factorization [28] (NMF) to decompose spatiotemporal patterns from complex darknet traffic. Scanner groups with the same spatiotemporal probing patterns were then highlighted. This method was able to detect all malware activities labeled by a human, and the study compared it with recent detection methods. Kanehara et al. [29] used nonnegative Tucker decomposition (NTD) [30] to extract interpretable co-occurrence scanning patterns of a coordinated group. NTD is an extension of NMF to higher-dimensional data. Kenehara et al. appended destination ports as a new axis and created an input tensor. Their system could detect orchestrated source IP addresses together with the target destination ports. Han et al. [50] finally integrated graphical lasso [25], NMF [27], and NTD [29] into an anomaly detection framework Dark-TRACER and achieved a 100% recall rate on darknet traffic data (up to /17 subnet scales) from October 2018 to 2020.

### C. BEHAVIOR-BASED APPROACHES

Gates [34] modeled various adversaries' scanning activities by understanding their incentives, benefits, and efficiency criteria. Then, she formulated the detection of multiple coordinating scanners as a set covering problem, which was a unique and intriguing approach. Her algorithm supposes that targets (destination IP addresses) of different scanners have few overlaps if these scanners cooperate. She finally found scanners that (1) cover enough IP address space and (2) have few target overlaps between different scanners. In a controlled environment, her algorithm achieved a low false positive rate in detecting horizontal and strobe scans. Bou-Harb et al. [35] performed heuristic clustering using five IP/TCP header fields to separate darknet traffic into independent probing flows. That research's unique assumption is that a probing flow from an orchestrated scanning campaign is predictable. The authors used a discrete Fourier transform (DFT) [36] for interpolation of a probing flow and a Kalman filter [37] to assess predictability. Their approach pinpointed a number of orchestrated probing campaigns. Torabi et al. [38] focused on the scanning objectives and behavioral characteristics to identify correlated and exploited IoT devices. The authors inferred the object of a scan from the unordered collection of scanned ports. The behavioral characteristics included ten features, including packet rate, number of source ports, and average length of IP packets. Their approach normalized the ten features and performed DBSCAN clustering to identify groups of IoT devices that behaved similarly. The detected groups were associated with well-known IoT malware and botnets by using the Shodan IoT search engine.[3] Pour et al. [39] proposed a probabilistic model to cleanse unrelated traffic through a pretreatment. The authors used binary classifiers based on convolutional neural networks or random forests to identify ongoing IoT botnets. Their framework uses agglomerative clustering to scrutinize distinctive network feature sets. It revealed a

[3]https://www.shodan.io/

momentous 440,000 compromised IoT devices and generated evidence-based artifacts related to 350 IoT botnets.

## III. METHODOLOGY

This section explains how to identify a fingerprint, i.e., a unique pattern of a scanning activity embedded in TCP/IP header fields. Section III-A outlines the procedure for identifying fingerprints, and Section III-B provides the formal definition and examples of fingerprints. The key process is detailed in Section III-C.

### A. IDENTIFYING FINGERPRINTS

Previous studies have revealed that some scan campaigns use packets that follow a unique pattern; we call the pattern a fingerprint. Because a scanning campaign manipulates many scanners, the pattern is expected to frequently appear in probing packets from many hosts. Therefore, we identify fingerprints in the following three steps.

1) Generate *TCP functions*
   A *TCP function* is a feature function that calculates a feature of a packet. We can produce TCP functions by combining TCP or IP header fields manually or by applying the method in our preliminary work [46] (see Appendix A).
2) Identify *effective signs*
   Suppose a feature function takes a specific value for many packets from many hosts. Then, the pair consisting of the corresponding TCP function and the value may be part of a fingerprint. We call such a pair an *effective sign*. We utilize two methods to identify effective signs. Our preliminary work [46] (see Section III-E) proposed the first method that regarded a pattern as a fingerprint if many packets had the pattern. The second method prioritizes the number of hosts over the number of packets that possesses a pattern. Section III-C describes the second method.
3) Unify *effective signs* into a fingerprint
   We combine several effective signs into a fingerprint. If two effective signs appear in packets from many hosts, the fingerprint is the pattern that both effective signs appear simultaneously. Otherwise, each effective sign becomes a fingerprint by itself.

### B. FINGERPRINT

This section provides the formal definitions of the terminology appearing in the previous section and enumerates well-known fingerprints. Let $\mathscr{P}$ be the set of all TCP packets and $\mathscr{B}$ be the set of all binaries. A *TCP function* $f : \mathscr{P} \to \mathscr{B}$ is a function that accepts a TCP packet and returns a binary. The set of all TCP functions is denoted by $\mathscr{F} := \{f \mid f : \mathscr{P} \to \mathscr{B}\}$. For example, $f(p) = \text{ip.src}$ is a TCP function that returns destination IP address of an input packet. *sign* $(f, b)$ is a pair consisting of a TCP function $f$ and a binary $b$. If a TCP packet $p$ satisfies $f(p) = b$, $p$ is said to have or satisfy a sign $(f, b)$. If many source IP addresses have a sign $(f, b)$, the sign is called an *effective sign*. Section III-C describes how to specify *effective signs*. For a certain packet, a sign $(f, b)$ is regarded as the proposition that has a true value if and only if the packet has the sign $(f, b)$. A *fingerprint* is a propositional formula

1) that is constructed using effective signs as proposition variables and $\{\wedge, \vee, \neg\}$ as logical operations
2) that becomes true for packets from many source IP addresses

where $\wedge$ means conjunction, $\vee$ disjunction and $\neg$ negation. A source IP address is said to have a fingerprint if the IP address sends "at least one packet" for which the fingerprint becomes true.

Some survey-objective scanners and malware embed unique patterns into packets, and fingerprints can express these patterns. For example, an IP address infected by Mirai [3] sends packets wherein a bitwise XOR between the sequence number and destination IP address equals zero. This pattern is represented by the following fingerprint:

$$\text{tcp.seq} \oplus \text{ip.dst} = 0. \tag{1}$$

The well-known fingerprints are summarized in Table 3. If a fingerprint named NAME becomes true for a packet, we call the packet NAME packet. For example, a Mirai packet is the packet that satisfies (1).

**TABLE 3. Fingerprints of well-known malwares and survey-objective scanners. $g_{\mathrm{L2B}}(\cdot)$ is a function that outputs the lower two bytes, and $\oplus$ is a bitwise XOR that performs the logical inclusive operation on each pair of corresponding bits.**

| Objective | Name | Fingerprint |
|---|---|---|
| Attack | Mirai [3] | $\text{ip.dst} \oplus \text{tcp.seq} = 0$ |
| | Hajime [47] | tcp.window=14600 |
| Survey | Massscan [48] | $g_{\mathrm{L2B}}(\text{ip.dst}) \oplus \text{tcp.dport} \oplus \text{ip.id} \oplus g_{\mathrm{L2B}}(\text{tcp.seq}) = 0$ |
| | ZMap [49] | ip.id = 54321 |

### C. IDENTIFYING EFFECTIVE SIGNS

This section describes how to identify effective signs $(f, b)$ given a TCP function $f$. An effective sign is a fingerprint component that appears in packets from many hosts.

The appearance ratio (frequency) is used to identify effective signs. Let $\mathscr{B}$ be the set of all binaries, $f$ be a TCP function, $P$ be a set of TCP packets, $S$ be a set of source IP addresses, and $P_s \subsetneq P$ $(s \in S)$ be the set of TCP packets originating from IP address $s$. Then, *appearance ratio* for a binary $b$ is defined as

$$r(b) := \frac{\#\{s \in S \mid \exists p \in P_s, f(p) = b\}}{\#S} \tag{2}$$

where $\#A$ is the cardinality of the set $A$. The appearance ratio means the proportion of source IP addresses that have the sign $(f, b)$. We define $\{b_n\}$ as the sequence of $f(P)$ in descending order of $r(b)$, where $f(P) := \{f(p) \mid p \in P\} \subseteq \mathscr{B}$. The subsequence from the $k$-th to the last element is defined as

$$\{b_{n \geq k}\} := (b_k, b_{k+1}, b_{k+2}, \cdots). \tag{3}$$

**TABLE 4.** TCP/IP header fields composed of TCP functions.

| | Field |
|---|---|
| IP header | ip.src, ip.dst, ip.id, ip.checksum |
| TCP header | tcp.seq, tcp.sport, tcp.dport, tcp.checksum, tcp.window, tcp.ugptr, tcp.ack |

Moreover, we define $r\left(\{b_{n\geq k}\}\right)$ as

$$r\left(\{b_{n\geq k}\}\right) := \left(r\left(b_k\right), r\left(b_{k+1}\right), r\left(b_{k+2}\right), \cdots\right). \quad (4)$$

The *effective indicator function* $e : \mathbb{N}^+ \to \mathbb{R}$ is defined as

$$e(k) := \frac{\sigma^2\left(r\left(\{b_{n\geq k}\}\right)\right)}{\sigma^2\left(r\left(\{b_{n\geq k+1}\}\right)\right)} \quad (5)$$

where $\mathbb{N}^+$ is the set of positive integers, $\mathbb{R}$ is the set of real numbers, and $\sigma^2(A)$ is the population variance of $A$. The effective indicator $e(k)$ represents the effect of $r\left(b_k\right)$ on $\sigma^2\left(r\left(\{b_{n\geq k}\}\right)\right)$. If $r(b_k)$ is much higher than $r\left(b_\ell\right)$ $(\ell \geq k+1)$, $e(k)$ takes on a large value. This implies that $(f, b_1), (f, b_2), \cdots, (f, b_k)$ are satisfied for packets from many more hosts than $(f, b_\ell)$ $(\ell \geq k+1)$.

Two parameters are used to identify effective signs, a positive integer $K$ and a positive real number $\alpha$. We calculate $k_{\max}$ as follows:

$$k_{\max} := \{1 \leq k \leq K \mid e(k) \geq \alpha\}. \quad (6)$$

Thus, all the effective signs using a TCP function $f$ are $(f, b_1), (f, b_2), \cdots, (f, k_{\max})$. If $k_{\max}$ does not exist, there are no effective signs.

## IV. EXPERIMENTAL SETTINGS

We applied our approach to darknet traffic and analyzed the fingerprints and the source IP addresses that had those fingerprints. Section IV-A describes the settings of our algorithm, and Section IV-B explains our dataset and computation time. Section IV-C deals with the preprocessing for analyzing fingerprints.

### A. PARAMETERS OF OUR APPROACH

Our experiment utilized all of the TCP functions expressed by taking a bitwise XOR between (a) 11 TCP/IP header fields in Table 4 and (b) the lower or higher two bytes of these fields. For example, the Mirai fingerprint has the ip.dst $\oplus$ tcp.seq as the TCP function (Table 3), and the Hajime one has tcp.window (Table 3). We specified effective signs for each TCP function and combined these effective signs into fingerprints represented by their logical conjunction. When Section III-C (Section III-E in [46]) identified effective signs, we set $K = 10$ (max_sign $= 10$) and determined $\alpha$ (sign_thres) so that at most 20 effective signs would be found in a day.

**TABLE 5.** Original and preprocessed darknet traffic data.

| | #ip.src | #packets |
|---|---|---|
| Original | 5,684,371 (100%) | 37,141,813,739 (100%) |
| Preprocessed | 1,295,374 (**22.8**%) | 37,021,340,731 (**99.7**%) |

### B. DARKNET DATA AND COMPUTATION TIME

A darknet, also known as a network telescope, passively monitors network traffic with an unreachable dark IP address block. It is an effective system for observing indiscriminate Internet-wide scans because it does not receive benign and regular network traffic. Our dataset consists of TCP SYN packets collected from a darknet operated by NICTER.[4] Because TCP SYN packets are used to survey active hosts and open ports [43], our experiment used only TCP SYN packets. NICTER's darknet has 298,280 IP addresses located worldwide. We applied our method to the TCP SYN packets collected from it over a period of one month, September 2021. The total number of TCP SYN packets was about 37.1 billion.

We implemented our approach day by day while removing fingerprints found so far on a fraction of the daily packets (the total number of packets during one month experiment is 5.4 billion), which were randomly selected, to reduce the computation time. Python and Julia language and 24 cores on a server AMD EPYC 7H12 (64 CPUs and 56 GB RAM) were utilized. The computation cost was approximately proportional to the number of packets and TCP functions, and the average daily computation time was 1.8 hours. We evaluated our results using all the packets (37.1 billion).

### C. PRETREATMENT FOR ANALYZING FINGERPRINTS

In network address translation (NAT) implementation, one or more local IP address is translated into one or more global IP addresses. In our experiments, a source IP address indicates a global IP address. We cleansed unrelated traffic by removing noise packets (i.e., misconfigured network traffic) and source IP addresses that sent these packets. Specifically, we removed any source IP address in which either (a) the number of destination IP addresses was less than 25 or (b) the number of packets was less than 30 throughout the entire experimental period. We also eliminated the packets from these IP addresses. For example, we removed a source IP address that sent ten packets from September 1st to 30th, 2022. The numbers of source IP addresses and packets before and after the pretreatment are summarized in Table 5. The number of source IP addresses decreased to 22.8%, but 99.7% of packets remained after the pretreatment. We regarded each remaining source IP address to be a *scanner*.

## V. EXPERIMENTAL RESULTS

We investigate the identified fingerprints in Section V-A and the proportions of each fingerprint packet for a scanner in

[4]Network Incident Analysis Center for Tactical Emergency Response: https://www.nicter.jp/en

**TABLE 6.** All fingerprints identified by our approach. In the objective column, "Attack" means that the fingerprint packet targets a few types of destination ports, whereas "Survey" means that the fingerprint packet scans various ports. We utilized two methods to identify effective signs. The first is the one proposed in our preliminary work (see Section III-E in [46]) and is denoted by "old"; the second is the one described in Section III-C and is indicated by "new". Date indicates the first day a method identified a fingerprint; blank means the method did not find the fingerprint during an experiment. The number of scanners denotes the source IP addresses that sent "at least one fingerprint packet" during the experimental period. "K", "M", and "B" mean thousand ($10^3$), million ($10^6$), and billion ($10^9$), respectively. **Bold numbers** means large numbers. #Scanner(/X) indicates the unique network number when we regard the first X bits of an IP address as a network. "#Packets/#Scanner" is the average packet number of a scanner, i.e., #Packets divided by #Scanner. The symbol $\wedge$ means conjunction, and $\oplus$ is bitwise XOR. The major destination port set is defined as the set of destination ports of the fingerprint packet. Some of the major destination port sets are defined in Table 7.

| Objective | Name | Method old / new | #Packets (%) #Scanner (%) | #Scanner (/24) #Scanner (/16) #Scanner (/8) | #Packets /#Scanner | Fingerprint | Major destination port set | Threat intelligence |
|---|---|---|---|---|---|---|---|---|
| Attack | Mirai | 1st / 1st | 1.69B (4.5%) 364K (**28.1**%) | 129,229 16,618 201 | 4,639 | ip.dst $\oplus$ tcp.seq $= 0$ | - subsets of {23,1023,2323} - {5555} | Mirai botnet [3] |
| | Hajime | 8th / 1st | 787.4M (2.1%) 245K (**18.9**%) | 84,441 12,231 198 | 3,208 | tcp.window $= 14600$ | - subsets of {23,80,8080} - {23,81} | Hajime botnet [47] |
| | Atk01 | / 1st | 60.2M (0.2%) 70.3K (5.4%) | 19,402 1,465 119 | 856 | tcp.window $= 29040$ | - Mozi01Pset - MiraiMoziPset | Mozi botnet [51] |
| | Atk02 | / 1st | 84.8M (0.2%) 55.1K (4.3%) | 7,070 1,613 137 | 1,540 | tcp.window $= 14520$ | - Mozi02Pset - MiraiMoziPset | Mozi botnet [51] |
| | Atk03 | 2nd / 2nd | 15.9M (0.0%) 976 (0.1%) | 926 404 53 | 16,291 | ip.dst $\oplus$ tcp.seq $= 3232235778$ $\wedge$tcp.window $= 1300$ | - {80},{81},{8000}, - {8080},{8081} | Papers [19], [38] |
| | Atk04 | 17th / | 1.5M (0.0%) 16 (0.0%) | 16 15 12 | 938K | tcp.seq $= 2018915346$ | {135} | CVE-2021-344527 [52] |
| | Atk05 | 1st / 16th | 1.5M (0.0%) 10 (0.0%) | 10 10 10 | 150 K | tcp.seq $= 333994513$ | Atk05Pset, {53,80,135,443} | Destination Ports 22 (SSH), 23 (Telnet), 53 (DNS),3389 (RDS), 80 and 443 (HTTP). |
| | Atk06 | 1st / 1st | 50.0M (0.1%) 8 (0.0%) | 8 7 5 | 6.25M | tcp.seq $= 30000$ $\wedge$tcp.window $= 65535$ | {22} | 22 (SSH) |
| Survey | Masscan | 1st / 1st | 21.3B (**57.6**%) 67.8K (5.2%) | 46,379 13,416 203 | 314K | $g_{L2B}$(ip.dst) $\oplus$ tcp.dport $\oplus$ ip.id $\oplus$ $g_{L2B}$(tcp.seq) $= 0$ | - {1433} - {445,1433} | Masscan [48] |
| | ZMap | 1st / 1st | 6.07B (**16.4**%) 54.8K (4.2%) | 35,426 11,754 200 | 111K | ip.id $= 54321$ | {22} | Zmap [49] |
| | Surv01 | / 1st | 48.0M (0.1%) 8,439 (0.7%) | 4,686 2,313 188 | 5,688 | tcp.window $= 0$ | Nothing | Nothing |
| | Surv02 | 3rd / | 17.9M (0.0%) 27 (0.0%) | 20 17 11 | 663K | tcp.seq $= 100 \wedge$ ip.id $= 123$ $\wedge$tcp.window $= 1024$ | {3389} | Nothing |

Section V-B1. Section V-B2 proposes a method to make scanner groups based on the proportions. Finally, we analyze these groups in Section V-B3 to V-B5.

### A. IDENTIFIED FINGERPRINTS

Table 6 present all the fingerprints identified by our approach (see Section III-A). Because many operating systems (OSes) have default window sizes (or other combinations of header fields known as defaults for OSes) for their SYN packets [45], [53], [54], [55], we excluded them from the analysis. Specifically, our approach found five fingerprint candidates with only default window sizes of 5840, 8192, 14600, 32120, or 65535 and excluded them except one with a window size of 14600. The window size of 14600 is a Hajime fingerprint and the default window size for some Android 4.4 devices. In this case, we regard the window size of 14600 as the Hajime fingerprint. Section V-B4 shows that the scanners with the Hajime fingerprint have major destination ports; hence, this treatment is rational.

We identified eight attack-objective and four survey-objective fingerprints of which eight fingerprints were previously unknown. The fingerprints include all the well-known fingerprints, Mirai, Hajime, Masscan, and ZMap. A fingerprint is said to be attack objective if the fingerprint packets are aimed at specific destination ports, whereas packets that are survey objective have a wide range of destination ports. About 74% of all packets come from well-known survey-objective scanner tools, Masscan and ZMap, but the number of these scanners is small. In contrast, 28.1% and 18.9% of all scanners have well-known attack-objective fingerprints, i.e., Mirai and Hajime, respectively.

The "old" method can identify fingerprints many packets have even if the number of scanners is small (Atk04–06 and Surv02); the "new" method can detect fingerprints that many scanners have even if the average number of packets per scanner (denoted by #Packets/#Scanner) is small (Mirai, Hajime, Atk01, Atk02, and Surv01). The difference derives from the algorithm design in which the old (new)

method regards a pattern as a fingerprint if many ''packets (scanners)'' have the pattern.

Fingerprints are distributed among various networks. For any fingerprint except for Atck01 and Atck02, the class C (/24) network has an average of less than 3 IP addresses (scanners). Atck01 and Atk02 have 3.6 and 7.8, respectively. In the case of Mirai, #Scanner / #Scanner(/24)=2.8. Regarding the class B (/16) network, attack-objective fingerprints are denser than survey-objective ones. Mirai, Hajime, Atk01, and Atk02 have an average of 20–50 IP addresses (scanners) in the class B network. In contrast, all survey-objective fingerprints and Atk03–Atk06 have 1–6 IP addresses (scanners).

Most fingerprints are simple enough to use a TCP function with only one field, and the tcp.window and tcp.seq are often included in fingerprints. Interestingly, the TCP function of the Atk03 fingerprint is the same as the Mirai one; hence, the adversary may have reused the scan tool of Mirai. The fingerprints of Atk03, Atk06, and Surv02 have several effective signs. The major destination port sets are examined in Section V-B4.

We have a survey-domain scanner list that is the set of IP addresses whose domains are well-known scanning entities (Shodan, Censys, etc.). We compared the scanners of each fingerprint with the list to determine whether or not a fingerprint was associated with a well-known scanning entity. Although we could not find a fingerprint of a well-known scanning entity, we observed a small overlap between the list and scanners of fingerprints for Masscan, ZMap, and Surv01, all of which are survey objectives. The domain of 0.7% of the scanners (IP addresses) with the Masscan fingerprint is Shadowserver Foundation, and 0.5% is Raid. The ZMap fingerprint has Shadowserver Foundation (0.9%), Raid (0.7%), BitSight (0.5%), and Cyber.Casa (0.5%). Surv01 has Shadowserver Foundation (5.9%), Alpha Strike Labs GmbH (4.9%), Raid7 (3.8%), Cyber.Casa (3.0%), BitSight (2.0%), Palo Alto Networks (1.1%), ONYPHE (0.6%), and Global Digital Network Plus (0.5%). Atk06 has Shadowserver Foundation (12.5%), which means one of eight Atk06 scanners has the domain of Shadowserver Foundation. That scanner sent one packet with the Atk06 fingerprint and the other 384,220 packets with no Atk06 fingerprint. Hence, it is not associated with the Atk06 fingerprint. The other fingerprints had well-known scanning entities of less than 0.5%. Note that some well-known scanning entities used various scanning tools. For example, Shadowserver Foundation used Masscan and ZMap tools. Another finding is that no fingerprint was associated with only one well-known entity.

*Remark 1: We identified all the well-known fingerprints (Mirai, Hajime, Masscan, and ZMap) and eight unknown fingerprints. The individual fingerprint was distributed among various networks. The improved method, denoted by "new" in Table 6, detected fingerprints of low-rate and coordinated scanners (Atk01, Atk02, and Surv01). Some scanning entities (such as Shadowserver Foundation and Rapid7) used various scanning tools.*

**TABLE 7. Major destination port sets.**

| Name | Destination port set |
|---|---|
| Mozi01Pset | {80,81,5555,7574,8080,8443,37215,49152,52869} |
| Mozi02Pset | Mozi01Pset ∪ {23, 1023, 2323} |
| MiraiMoziPset | Mozi01Pset ∪ {8081, 8181, 60001} |
| Atk05Pset | {22,23,53,80,135,443,3389} |

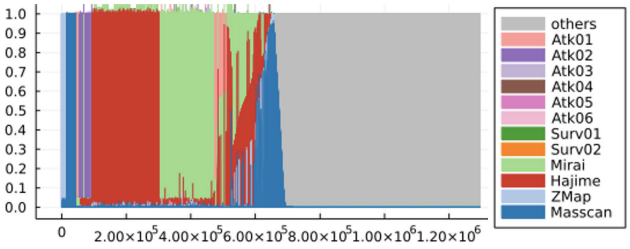


**FIGURE 1. Proportion of each fingerprint packet relative to total number of packets for each scanner.**

### B. FINGERPRINT-BASED SCANNER's GROUPING

#### 1) SCANNER's FINGERPRINT RATIO

We calculated the proportion of each fingerprint packet relative to the total number of packets for each scanner. Fig. 1 portrays the proportions, where the horizontal axis represents scanners and the vertical axis represents the proportion of each fingerprint packet. The legend corresponds to the identified fingerprints in Table 6. Because one packet can have multiple fingerprints, the sum of proportions can exceed one. ''Others'' colored gray means the proportion of packets with no fingerprints. The figure indicates that about half of the scanners mainly sent fingerprint packets. 49.4% of the scanners sent at least 30 packets with a fingerprint, and the proportion of the fingerprint packet exceeds 33%. We called the scanners *fingerprint scanners*. We also discovered that 81.2% of the packets have at least one fingerprint. Thus, the fingerprints helped us to analyze 81.2% of the packets and 49.4% of the scanners. Moreover, we found that Mirai and Hajime packets occupied a large proportion in many of the scanners. In particular, 13.1% of the scanners sent at least 30 Mirai packets, and the proportion of Mirai (hereafter, Mirai proportion, etc.) was 95% or more, while 15.8% sent at least 30 Hajime packets, and the Hajime proportion was 95% or more. Summing up the two fingerprints, 29.4% of the scanners sent 30 packets with Mirai or Hajime fingerprints, and the proportion of packets with Mirai or Hajime was 95% or more.

*Remark 2: We could analyze 81.2% of the packets and 49.4% of the scanners by fingerprints. 29.4% of scanners predominantly sent Mirai or Hajime fingerprint packets.*

#### 2) GROUPING SCANNERS

Fig. 1 indicates that the proportions of each fingerprint packet have various patterns. We decided to group *fingerprint scanners* on the basis of these proportions and in accordance

**TABLE 8. Scanner groups based on proportion of each fingerprint packet. #Scanner(/X) indicates a unique network number when we regard the first X bits of an IP address as a network.**

| Group name | Objective | #Scanner (%) § | #Scanner (/24) | #Scanner (/16) | #Scanner (/8) | Affiliation condition | Major fingerprints |
|---|---|---|---|---|---|---|---|
| pure-Hajime | Attack | 204,794 (**32.0%**) | 73,667 | 10,992 | 195 | No.1, Hajime-R$\geq$ 0.95 $\wedge$ Hajime-P $\geq$ 30 | Hajime |
| pure-Mirai | Attack | 170,091 (**26.6%**) | 100,178 | 15,543 | 200 | No.1, Mirai-R$\geq$ 0.95 $\wedge$ Mirai-P $\geq$ 30 | Mirai |
| pure-Atk02 | Attack | 37,848 (5.9%) | 3,608 | 1,049 | 108 | No.1, Atk02-R$\geq$ 0.95 $\wedge$ Atk02-P $\geq$ 30 | Atk02 |
| pure-Massscan | Survey | 26,175 (4.1%) | 17,939 | 6,010 | 199 | No.1, Masscan-R$\geq$ 0.95 $\wedge$ Masscan-P $\geq$ 30 | Masscan |
| pure-ZMap | Survey | 16,215 (2.5%) | 7,840 | 2,593 | 193 | No.1, ZMap-R$\geq$ 0.95 $\wedge$ ZMap-P $\geq$ 30 | ZMap |
| pure-Atk01 | Attack | 14,816 (2.3%) | 9,016 | 1,241 | 84 | No.1, Atk01-R$\geq$ 0.95 $\wedge$ Atk01-P $\geq$ 30 | Atk01 |
| pure-Surv01 | Survey | 3,018 (0.5%) | 2,622 | 1,337 | 170 | No.1, Surv01-R$\geq$ 0.95 $\wedge$ Surv01-P $\geq$ 30 | Surv01 |
| pure-Atk03 | Attack | 930 (0.1%) | 885 | 390 | 49 | No.1, Atk03-R$\geq$ 0.95 $\wedge$ Atk03-P $\geq$ 30 | Atk03 |
| pure-Surv02 | Survey | 26 (0.0%) | 19 | 16 | 11 | No.1, Surv02-R$\geq$ 0.95 $\wedge$ Surv02-P $\geq$ 30 | Surv02 |
| pure-Atk06 | Attack | 7 (0.0%) | 7 | 6 | 4 | No.1, Atk06-R$\geq$ 0.95 $\wedge$ Atk06-P $\geq$ 30 | Atk06 |
| pure-Atk04 | Attack | 5 (0.0%) | 5 | 5 | 4 | No.1, Atk04-R$\geq$ 0.95 $\wedge$ Atk04-P $\geq$ 30 | Atk04 |
| pure-Atk05 | Attack | 4 (0.0%) | 4 | 4 | 4 | No.1, Atk05-R$\geq$ 0.95 $\wedge$ Atk05-P $\geq$ 30 | Atk05 |
| Atk01-Mirai | Attack | 54,116 (8.5%) | 12,036 | 460 | 55 | No.2, Atk01-R$\geq$ 0.2 $\wedge$ Atk01-P$\geq$ 30 | Atk01, Mirai |
| Atk02-Mirai | Attack | 14,660 (2.3%) | 3,611 | 754 | 102 | No.3, Atk02-R$\geq$ 0.1 $\wedge$ Atk02-P$\geq$ 30 | Atk02, Mirai |
| Hajime-Mirai | Attack | 25,808 (4.0%) | 9,295 | 2,554 | 177 | No.4, Hajime-R$\geq$ 0.25 $\wedge$ Hajime-P$\geq$ 30 | Hajime, Mirai |
| others* | | 75,256 (11.8%) | | | | | Mirai, others |

* Scanners not belonging to any group.
§ Proportion of group scanners relative to total number of scanners.

with two policies: (a) first, we grouped the scanners having only one fingerprint; then we grouped those having several fingerprints; (b) we successively made groups while removing scanners that already had been assigned to a group. Table 8 shows the resulting groups. In the table, ''Attack'' indicates a scanner group targeting specific destination ports, and ''Survey'' indicates a group that sends packets to a wide range of destination ports. Section V-B3 provides evidence supporting the classification results of the group objectives. The ''affiliation condition'' in Table 8 indicates a pair consisting of ''No.'' and a proposition formula that decides whether a scanner belongs to the group. ''No.'' indicates the creation order, and ''No.M'' means that the group was created after removing scanners in groups No.1 to No.M-1. The proposition formula represents the requirement for both fingerprint packets and those proportions. For example, the condition ''Hajime proportion is 95% or more, and Hajime packets number 30 or more'' is denoted by ''Hajime-R $\geq$ 0.95 $\wedge$ Hajime-P $\geq$ 30''. The suffix ''-R'' means ratio, and ''-P'' means packets. We call a group with mostly one fingerprint a *pure group* (such as pure-Hajime, pure-Mirai, etc.) and a group with several fingerprints a *mixed* group (Atk01-Mirai, Atk02-Mirai, and Hajime-Mirai). The proposition formula of a pure group is ''the number of the fingerprint packet is 30 or more, and the fingerprint proportion is 0.95 or more''. Therefore, a pure group consists of scanners that frequently send packets with a specific fingerprint. We made groups in the order of pure groups (such as pure-Hajime, pure-Mirai, etc.) followed by mixed groups (Atk01-Mirai, Atk02-Mirai, and Hajime-Mirai) and assigned the remaining scanners to the ''others'' group.

The above procedure produced twelve pure groups that had the same numbers of fingerprints. The mixed groups were Atk01-Mirai, Atk02-Mirai, and Hajime-Mirai, and their proportions of each fingerprint packet are illustrated in Fig. 2. More than half of the scanners in the Hajime-Mirai group mainly sent Hajime and Mirai packets. Hence, those scanners were suspected of being infected by both Hajime and Mirai botnets. Both malware may have infected a device. Another case is where the scanner may have several devices, and different botnets (such as Hajime, Mirai, etc.) may have infected different devices in the scanner.

As with the case of the fingerprints' network distribution, the individual scanner groups were distributed over a wide range of networks. For any fingerprint except pure-Atk02, the class C (/24) networks had an average of less than 5 IP addresses (scanners). pure-Atk02 had 10.5 IP addresses. Atk01-Mirai had an average of 117.6 IP addresses for a class B (/16) network, while any other group has an average of less than 37 IP addresses.

*Remark 3: Some scanner groups mostly sent only one type of fingerprint packet, but others sent multiple types. Individual scanner groups were distributed over a wide range of networks.*

### 3) SCAN OBJECTIVES OF SCANNER GROUPS

We classified the scanner groups into attack or survey objectives. An attack-objective group intensively sends packets to a few destination ports, while a survey-objective group sends packets to extensive destination ports.

We developed a statistic for deciding the objectives of scanner groups as follows. We defined a *top-k port* as a destination port with the $k$-th largest number of packets. Fig. 3

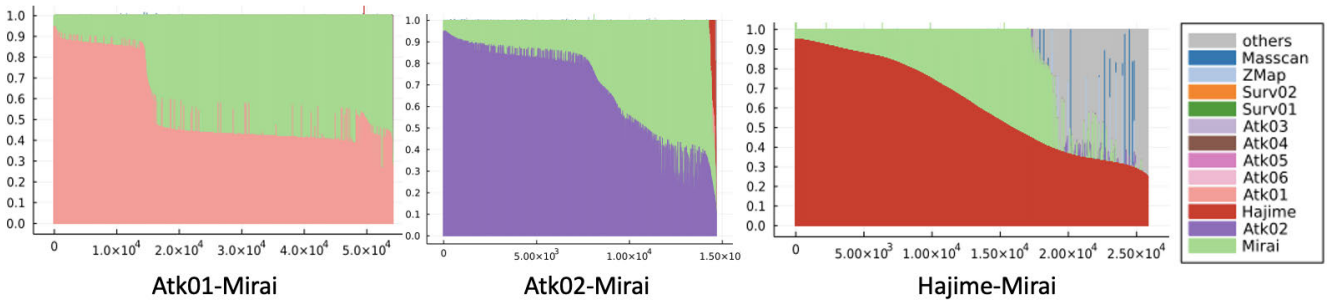


**FIGURE 2.** Proportions of each fingerprint packet in mixed groups.

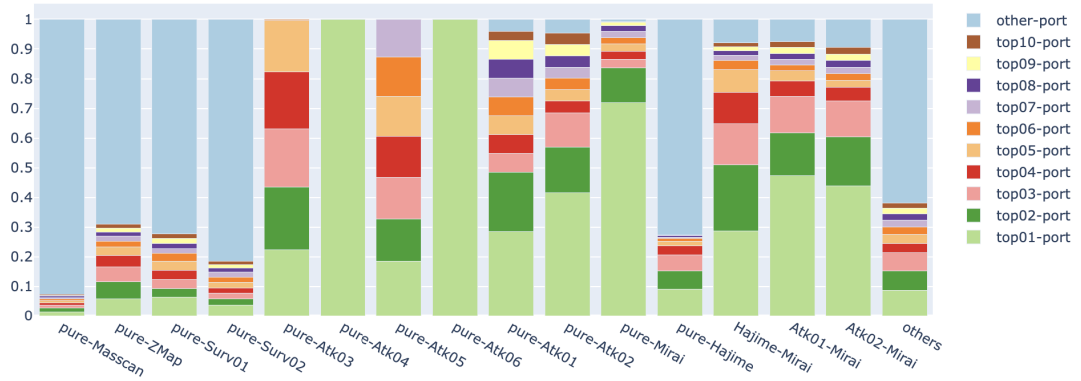


**FIGURE 3.** Proportion of top-*k* ports in each scanner group.

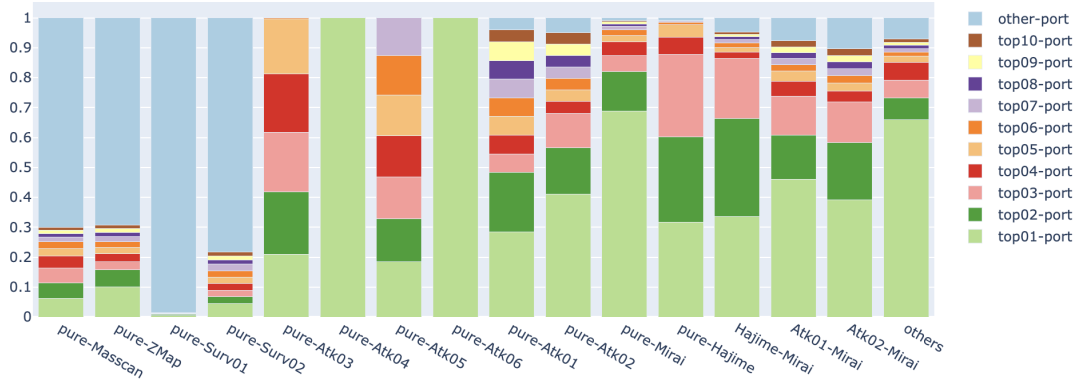


**FIGURE 4.** Proportion of top-*k* ports in each scanner group when we remove scanners whose packets number is in the top 5%.

shows the proportions of the top-1 to top-10 ports and the other ports. In the pure-Hajime group, which is related to the malware Hajime, the other ports occupy a large proportion. This result contradicts the assumption that a group with an attacking objective intensively sends packets to a few destination ports. Here, we suspected that a few large-scale survey-objective scanners incessantly send packets with the Hajime fingerprint. The following investigation confirms this hypothesis: 0.1% of the scanners (253 source IP addresses) in the pure-Hajime group sent packets to more than 60,000 destination ports, and these packets amounted to 70.9% of those from the pure-Hajime group. These large-scale survey-objective scanners significantly affect the proportions of top-*k* ports. To mitigate their impact, we removed scanners whose packet number was in the top 5%. Fig. 4 illustrates the proportions of top-*k* ports for each scanner group after this preprocessing. In the pure-Hajime group, the top-1 to top-3 ports occupy 90%, which implies that the pure-Hajime group has an attacking purpose. The ''others'' group shows the same situation: ''other port'' occupies a large proportion before the preprocessing (Fig. 3) but a small proportion after the preprocessing (Fig. 4). The preprocessing had little

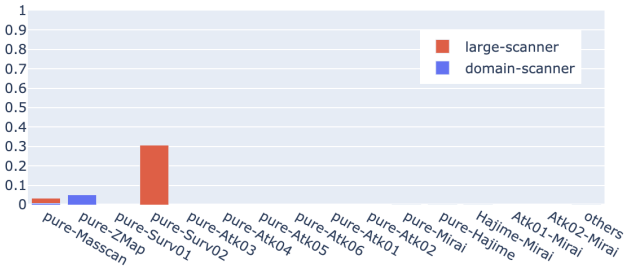


**FIGURE 5.** Proportion of IP address overlap between a scan group and survey-objective scanner lists.

effect on the proportions of top-$k$ ports for the other groups. Here, group is said to have an attack objective if the sum of proportions of top-1 to top-10 ports is 50% or more in Fig. 4. Otherwise, the group has a survey objective. This definition does not contradict the classification results of the well-known malwares and scan tools; that is, malware Mirai and Hajime are attack-objective, and scan tools Masscan and ZMap are survey-objective. The classification results are listed in the "Objective" field of Table 8.

To ascertain the reliability of the classification results, we examined the IP address overlap between the scan groups and survey-objective scanner lists (Fig. 5). In the figure, "large-scanner" indicates that the IP address satisfies two conditions: (1) there are 30 or more unique destination ports in a day, and (2) the number of packets exceeds the number of our darknet's IP addresses on the same day. 1,345 IP addresses satisfied both conditions on at least one day during the experimental period. "domain-scanner" indicates that an IP address is considered a survey-objective scanner because of the domain name (e.g., Shodan, Censys). The y-axis indicates the ratio of the intersection of a group and each scanner list to a group. Three out of the four survey groups, i.e., pure-Masscan, pure-ZMap, and pure-Surv02, show overlap with the survey-objective scanner lists, while the other groups show hardly any overlap. This result is evidence of the correctness of the group classification.

*Remark 4: We classified scanner groups into attack or survey objectives where the attack-objective group intensively targets a few ports, while the survey-objective group scans a wide range of ports. We verified the validity of the classification results with well-known fingerprints' objectives (Table 3) and our survey-objective scanner lists.*

#### 4) DESTINATION PORT SETS OF SCANNER GROUPS

In recent years, some botnets have scanned several ports and aimed at multiple vulnerabilities; hence, investigation of destination port sets has become more important. This study defines a *destination port set* as the set of destination ports of a scanner in one day. We found that the attack-objective fingerprint packets have major destination port sets. Fig. 6 illustrates the proportions of destination port sets of fingerprint packets for each pure group. Here, $s_{i,d}$ is the destination port set of a scanner $i$ on day $d$. Then, the proportion of a

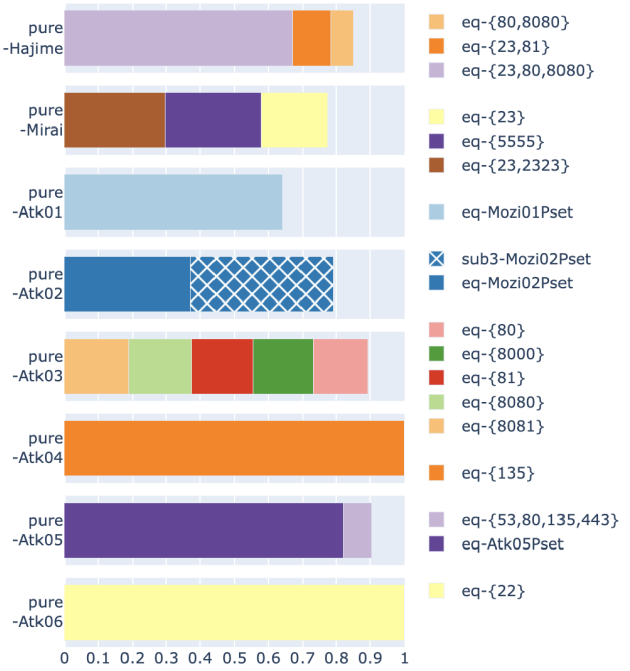


**FIGURE 6.** Proportion of destination port sets of fingerprint packets from pure groups. The destination port sets "Mozi01Pset", "Mozi02Pset", and "Atk05Pset" are defined in Table 7.

destination port set $s$ is

$$\frac{\sum_{i,d} \#\{s_{i,d} \mid s_{i,d} = s\}}{\sum_{i,s',d} \#\{s_{i,d} \mid s_{i,d} = s'\}}. \tag{7}$$

$d$ can be any day in the experimental period, while $i$ is restricted to a scanner group and the destination port set $s_{i,d}$ is made from only the corresponding fingerprint packets. In the pure-Hajime group, 67.2% of the destination port sets are $\{23, 80, 8080\}$, 11.2% are $\{23, 81\}$, and 6.7% are $\{80, 8080\}$. In the legend, "eq-PORTSET" means the port set PORTSET, and "sub{$N$}-PORTSET" represents the family set that consists of the proper subset of the PORTSET whose size is equal to or larger than the size of PORTSET minus $N$. For example, sub2-$\{1, 2, 3\}$ = $\{\{1, 2\}, \{2, 3\}, \{3, 1\}, \{1\}, \{2\}, \{3\}\}$. We define *the proportion of a family set* to be the sum of proportions over the sets of the family set. Note that we defined "sub{$N$}-" in order to speculate about the destination port sets for the IP address space, not our darknet IP addresses. Because our darknet is a portion of the IP address space, we guess that a port set that is included in "sub{$N$}-PORTSET" is "PORTSET" in the entire IP address space. For example, we regard that Mozi02Pset occupies 79.0% for the Atk02 group concerning the IP address space, where Mozi02Pset is defined in Table 7. Mozi02Pset is large; hence, the pure-Atk02 group probably targets many vulnerabilities. In addition, all scanners belonging to pure-Atk04 and pure-Atk06 send fingerprint packets to only one port. A few types of destination port set account for more than 67% for any pure

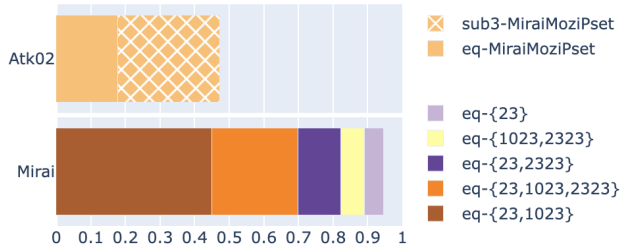


**FIGURE 7.** Proportions of destination port sets of Atk02 and Mirai fingerprint packets from the Atk02-Mirai group. The destination port set "MiraiMoziPset" is defined in Table 7.

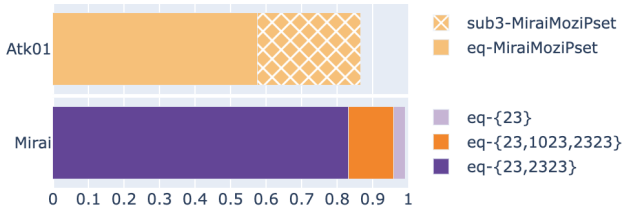


**FIGURE 8.** Proportions of destination port sets of Atk01 and Mirai fingerprint packets from the Atk01-Mirai group. The destination port set "MiraiMoziPset" is defined in Table 7.

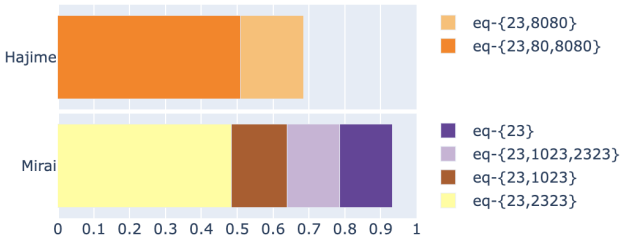


**FIGURE 9.** Proportions of destination port sets of Hajime and Mirai fingerprint packets from the Hajime-Mirai group.

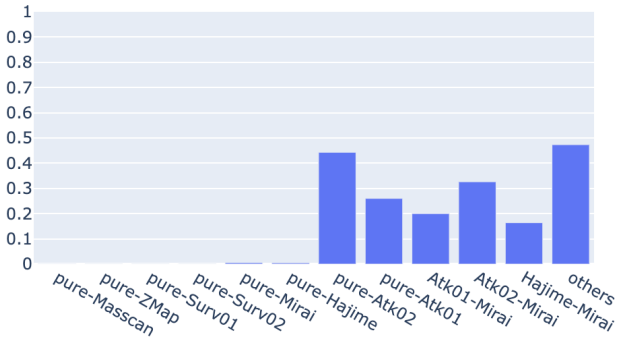


**FIGURE 10.** Mozi infection rates of each scanner group.

attack group. Hence, these groups probably target specific vulnerabilities.

Next, let us investigate the destination port sets of the mixed groups. Fig. 2 indicates that scanners in the Atk02-Mirai group mainly send Atk02 or Mirai fingerprint packets. Fig. 7 illustrates the proportions of destination port sets of each fingerprint. The major destination port sets of the Mirai packets consist of 23/TCP, 1023/TCP, and 2323/TCP. The ports included in the major destination port sets of Atk02 and Mirai are different, where MiraiMoziPset, the major destination port set of Atk02, is defined in Table 7. The above results mean that if we use all types of fingerprint packets, we would analyze the union of the destination port sets of Mirai and Atk02 packets and thereby mistake several attack intentions for one intention, but if we analyze the destination port sets for each fingerprint, then we can distinguish the different attack intentions. Additionally, Fig. 7 and Fig. 8 indicate that different fingerprints, Atk01 and Atk02, have the same major destination port. The TCP functions of Atk01 and Atk02 are the same (see Table 6). From the two facts, we deduce that an adversary may have reused the Atk01 fingerprint to make Atk02 or vice versa.

Finally, we tried to determine whether fingerprint packets' major destination port sets varied if we restricted these packets to packets from a scanner group. The destination port sets of all fingerprint packets (Mirai, Hajime, Atk01, and Atk02) are almost identical for different scanner groups. For example, Fig. 6, Fig. 7, and Fig. 8 indicate that the major destination port sets of Mirai packets are mostly subsets of $\{23, 1023, 2323\}$, while the pure-Mirai group has an additional destination port set $\{5555\}$. The major destination port sets of Hajime packets are the subsets of $\{23, 80, 8080\}$ for both the pure-Hajime and Hajime-Mirai groups (Fig. 6 and Fig. 9). The major destination port set of Atk01 packets is Mozi01Pset in the pure-Atk01 group (Fig. 6) and MiraiMoziPset in the Atk01-Mirai group (Fig. 8), where MiraiMoziPset $=$ Mozi01Pset $\cup$ $\{8081, 8181, 60001\}$ as shown in Table 7. Table 6 summarizes the major destination port sets of fingerprint packets.

*Remark 5: Every attack-objective fingerprint packet had unique major destination port sets, which did not vary fundamentally, even if the packets were restricted to the packets from any scanner group.*

### 5) FINGERPRINTS AND DESTINATION PORT SETS OF THE MOZI BOTNETS

The IBM Security X-Force discovered that the Mozi botnet accounted for 90% of the total IoT network traffic in September 2020 [51]. This botnet evolved from the source codes of infamous malware families such as Mirai, IoT Reaper, and Gafgyt. Mozi increased the IoT attack volume from October 2019 to June 2020 by 400% compared with the total IoT attack cases in the previous two years. In this section, we revealed that (1) both Atk01 and Atk02 are probably fingerprints of the Mozi botnet; (2) the fingerprint packets have the major destination port sets, i.e., Mozi01Pset, Mozi02Pset, and MiraiMoziPset, defined in Table 7. We obtained 286,339 IP addresses suspected of being infected by the Mozi botnet by connecting the peer-to-peer network of the Mozi as a dummy host. We believe that this Mozi IP address list contains 10–20% of all Mozi-infected IP addresses in the report by NICTER.[5]

[5]https://www.nicter.jp/en

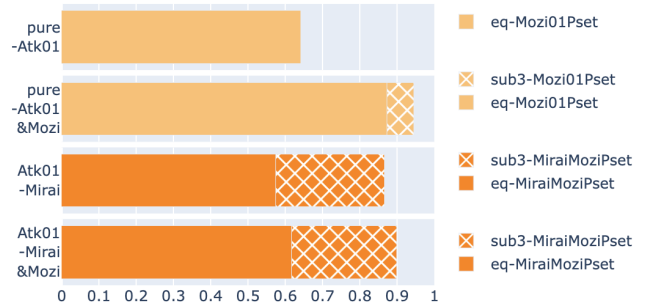


**FIGURE 11.** Proportion of destination port sets of the Atk01 fingerprint packets for each scanner group. "&Mozi" means that scanners are restricted to those in the Mozi list. "Mozi01Pset" and "MiraiMoziPset" are defined in Table 7.

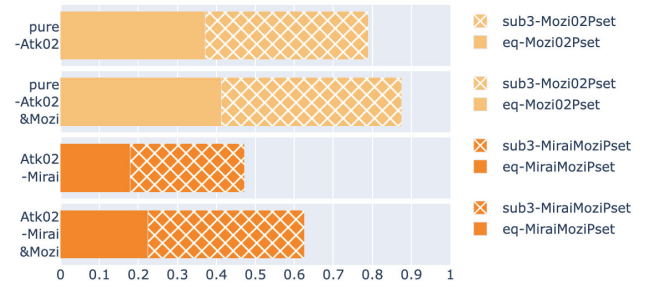


**FIGURE 12.** Proportion of destination port sets of the Atk02 fingerprint packets for each scanner group. The "&Mozi" means that scanners are restricted to those in the Mozi list. "Mozi02Pset" and "MiraiMoziPset" are defined in Table 7.

Fig. 10 shows the Mozi infection rates of each scanner group. The infection rate is the intersection between a scanner group and the Mozi list divided by the scanner group. None of the survey groups (pure-Masscan, pure-ZMap, pure-Surv01, and pure-Surv02) are included in the Mozi list. The pure-Mirai and pure-Hajime groups each have tiny proportions, less than 1%. In contrast, the scanner groups that send Atk01 or Atk02 packets have proportions of 20–44% and the Hajime-Mirai scanner group has 16.4%. Therefore, we suppose that Atk01 and Atk02 are related to the fingerprint of Mozi. We found the major destination port sets of the Mozi by restricting the scanners to those of the Mozi list for the pure-Atk01, pure-Atk02, Atk01-Mirai, and Atk02-Mirai groups. In the pure-Atk01 group illustrated in Fig. 11, the proportion of Mozi01Pset becomes 87.6% when the scanners are restricted to those of the Mozi list, which is higher than 64.1% when all scanners in the group are included. This result suggests that Atk01 is a Mozi fingerprint and its major destination port set is Mozi01Pset. We came to similar conclusions about pure-Atk02, Atk01-Mirai, and Atk02-Mirai (Fig. 11 and Fig. 12). In summary, we identified two fingerprints of Mozi botnets, Atk01 and Atk02, each of which has two major destination port sets (Table 9).

We tried to determine whether the Hajime-Mirai and "others" groups' scanners, which overlapped the Mozi list (Fig. 10), had the features of Mozi botnets. Fig. 2 indicates

**TABLE 9.** Fingerprints, major destination port sets, and scanner groups of the Mozi botnets. The major destination port sets are defined in Table 7.

| Fingerprint | Major destination port sets | Scanner group |
|---|---|---|
| Atk01 | Mozi01Pset<br>MiraiMoziPset | pure-Atk01<br>Atk01-Mirai |
| Atk02 | Mozi02Pset<br>MiraiMoziPset | pure-Atk02<br>Atk02-Mirai |

that the Hajime-Mirai group mainly sends Hajime or Mirai packets. The major destination port sets of these packets do not differ between before and after restricting the scanners to the Mozi list, which indicates that the Hajime-Mirai group has no features of the Mozi botnet. In the "others" group, the major destination port sets of "fingerprint packets" show no distinctive features after restricting the scanners to those on the Mozi list. In contrast, the destination port sets of "all packets (not restricted to fingerprint packets)" after restricting the scanners to those on the Mozi list are similar to the Mozi port sets defined in Table 9. These similar destination port sets have many variations, and none of them make up a large proportion. We guess an unrevealed scanner group scanned fixed ports similar to the Mozi ones and other random ports in the same strategy as the HNS botnet [56].

*Remark 6: We identified two fingerprints of Mozi botnets, Atk01 and Atk02, each of which has two major destination port sets (Table 9).*

### C. ASSOCIATING FINGERPRINTS WITH THREAT INTELLIGENCE

This section tries to validate the found fingerprints' reliability by associating fingerprints with thread intelligence. Because Section V-B5 demonstrated that Atk01 and Atk02 seemed to be Mozi fingerprints, this section investigates Atk03–06, Surv01, and Surv02 and summarizes the results in Table 6.

The major destination ports of Atk03 are 80, 81, 8000, 8080, and 8081, all of which are associated with HTTP. Cohen et al. [19] detected the source IP addresses targeting these ports and named these IPs Cluster B. Torabi et al. [38] noticed that ports 80, 81, 8000, and 8080 were targeted by a relatively large number of compromised devices despite occupying a small portion of darknet packets. Atk03 is probably related to these two scanner groups.

On Tuesday, July 6, 2021, Microsoft issued CVE-2021-34527 regarding a Windows Print Spooler vulnerability [52]. It affects all versions of Windows and gives a low-privilege user (attacker) code execution with admin privileges on the machine. Because 135/TCP is a target port for the attacker, Atk04 might be associated with the attack.

Although we can neither associate Atk05 nor Atk06 with any attack campaigns or information-security vulnerabilities, these fingerprints target major vulnerable ports 22 (SSH), 23 (Telnet), 53 (DNS), 3389 (remote desktop service),

80 (HTTP), and 443 (HTTP). We can neither connect survey-objective fingerprints Surv01 and Surv02 to any scanning campaigns.

## VI. DISCUSSION

### A. ADVANTAGES OF PROPOSED APPROACH

We proposed a fingerprint identifier to detect coordinated scanning campaigns. This section describes the benefits of a fingerprint-based approach compared with port-based, time-series, and behavior-based methods. Then, we explain the superiority of our approach over the other fingerprint-based approaches.

The greatest advantage of the fingerprint-based approach is that it identifies coordinated scanning with reliable evidence. For example, Atk03 in Table 6 is an unpublished fingerprint in which a bitwise XOR between the destination IP address and the sequence number takes a specific value. Because the sequence number is usually a random value, the fingerprint appears with a probability of $1/2^{32}$ (the sequence number is represented by 32 bits). Therefore, many scanners and packets that have the fingerprint indicate that these scanners utilize identical scanning software for a specific purpose. In contrast, port-based, time-series, and behavior-based approaches detect orchestrated scanners using their original indicators and parameters. Although these indicators attempt to quantify the similarity or synchronization of scanning behavior, their digitalized values are difficult to interpret. Hence, tuning the parameters is hard, and we can not recognize to what extent the detected scanners cooperate. Thus, unrelated scanners wrongly belong to a scanner group with some probability. Another merit of the fingerprint-based approach is it captures individual scanning groups. Different adversaries use distinct fingerprints even if they exploit an identical vulnerability. Therefore, we can recognize each scanning group by identifying fingerprints. We should note that the port-based approach has trouble distinguishing individual groups because groups with identical target services scan the same ports.

To the best of our knowledge, our approach is the first method to identify flexible fingerprints (represented by TCP/IP headers and operations on them) possessed by low-rate coordinated scanners. Previous research [3], [41] created fingerprints from open-source codes. Another study [43] identified fixed-form fingerprints. However, none of these methods can discover flexible fingerprints. Our preliminary work [46] devised an algorithm for identifying flexible fingerprints, but the algorithm overlooked scanning campaigns if scanners perform low-rate scanning activity. This paper developed an algorithm and detected low-rate scanning activity, such as the Mozi botnet.

*Remark 7: The greatest advantage of the fingerprint-based approach is that it identifies coordinated scanning "with reliable evidence." Our method is the first attempt to identify flexible fingerprints that (1) are represented by TCP/IP headers and "operations" (such as bitwise XOR) on them; (2) are possessed by low-rate coordinated scanners.*

### B. APPLICATIONS

One application of our approach is to change the input data. Although we are currently utilizing only darknet traffic, we may obtain more reliable fingerprints if we use traffic originating from infected devices. Srinivasa et al. [57] identified 11,119 IP addresses that attacked their honeypot and network telescope. The probe packets from these sources are pure and would show explicit characteristics of adversaries. Our approach could easily discover and reinterpret these characteristics as fingerprints. Another data source is live servers. Richter and Berger [58] tracked unsolicited traffic captured at the firewalls of some 89,000 hosts of a major Content Distribution Network (CDN) that (1) were distributed across some 1,300 networks and (2) offered services and thus emitted traffic. They revealed that conventional darknets could only partially illuminate scanning activity and might severely underestimate widespread attempts to scan and exploit individual services in specific prefixes or networks. We should obtain new insights by applying our approach to the dataset.

Fingerprints would assist a honeypot in deluding and interconnecting with bonnets. Safaei Pour et al. [59] proposed HoneyComb that behaves like a honeypot on a large darknet network. Their research centered on Mirai and its variants; hence, HoneyComb crafted Mirai's deceiving packets (i.e., TCP SYN-ACK packets) to delude and interconnect with Mirai-infected IoT devices. Making deceiving packets of various botnets is a challenging task that hinders the extension of HoneyComb. Fingerprints contribute to creating forged packets.

### C. LIMITATIONS

Although fingerprints are the most reliable feature in identifying scan campaigns, some shortcomings exist. We could not discover fingerprints for 50.6% of the scanners and 18.8% of the packets, as demonstrated in Section V-B1. There are two possibilities for that.

First, we have yet to find more complex fingerprints. In our experiments, we used only bitwise XOR as the bitwise operation; hence, we could not find fingerprints using other operations, such as bit shifts. Although we can incorporate any bitwise operation into a TCP function, the number of TCP functions increases as we include more bitwise operations. Because the computation time of our algorithm increases at a rate proportional to the number of TCP functions, we need to restrict their variegation. The restriction leads to losing the flexibility of TCP functions and fingerprints.

The other reason we could not uncover fingerprints is that not all scanners provide fingerprints. A fingerprint is an auxiliary tool for low-cost re-identification of a probe packet response, and scanners do not have to possess a fingerprint. Therefore, we should use some of the other techniques introduced in Section II (port-based, time-series, and behavior-based approaches) to detect coordinated scanning hosts after eliminating fingerprint packets and scanners.

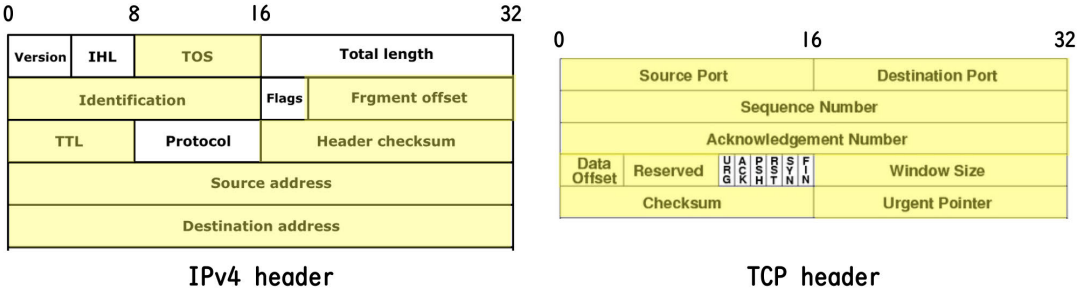


**FIGURE 13.** Header fields of the IPv4 and TCP protocols. Modifiable fields are in yellow, and unchangeable fields are in white.

## VII. CONCLUSION

We improved a fingerprint identifier to enable it to detect low-rate and well-coordinated scanning activities. The improved method discovered these stealth scanners (Atk01, Atk02, and Surv01) and all the well-known fingerprints (Masscan, ZMap, Hajime, and Mirai) on one month's worth of darknet data collected from about 300K unused IP addresses. We grouped scanners based on the proportions of the fingerprint packets. Then, we revealed the intention of each scanner group by scrutinizing the destination port sets. Most groups had major destination port sets; hence, we can speculate on the target services and vulnerabilities. A significant discovery is identifying the fingerprints of the Mozi botnet and its major destination port sets, all of which were previously unknown. We also found the source IP addresses suspected of being infected by several botnets. We associated fingerprints with threat intelligence and verified their reliability. In the future, we want to reveal the OSes of the scanning machines to understand which OSes are indiscriminately targeted and infected by malware (e.g., botnets or worms). We will also discover more reliable fingerprints by using our algorithm to probe packets captured from dynamic malware analysis. Finally, we aim to build a system capable of automatically associating identified fingerprints with open-source threat intelligence [60].

## APPENDIX A GENERATING TCP FUNCTIONS BASED ON GENETIC ALGORITHM

We define a *TCP function* as a function that accepts a TCP packet and returns a binary, and the TCP function is a fingerprint component. This section describes the method that our preliminary work [46] proposed to generate TCP functions. A new TCP function is generated from the current TCP functions. Section VII-A shows the TCP/IP header fields used in the TCP functions. Section VII-B defines the initial TCP functions, and Section VII-C describes how to make a new TCP function. Section VII-D presents an analogy between the genetic algorithm (GA) and our method.

### A. HEADER FIELDS USED IN FINGERPRINTS

As shown in Fig. 13, the header fields of the IPv4 and TCP protocols are categorized into (a) modifiable fields (yellow) and (b) unchangeable fields (white). Modifications to unchangeable fields prevent IPv4 or TCP from working correctly. For example, a destination host can not parse a packet if the IPv4 version is wrong. Therefore, attackers do not take unchangeable fields into the TCP functions.

### B. INITIALIZING TCP FUNCTIONS

We define the *initial TCP functions* $F_1 \subseteq \mathscr{F}$ as the set of *TCP functions* that returns modifiable header fields (colored yellow in Fig. 13). However, some header fields have the same binary value for almost all packets. We eliminated the corresponding *initial TCP functions* because the outputs of these functions are indistinguishable and hence inappropriate for components of fingerprints.

### C. GENERATING TCP FUNCTION

Given a set of TCP functions $F \subseteq \mathscr{F}$, our algorithm repeatedly generates a new TCP function and adds it to the set. (a) *feature extraction* and (b) *binary operation* are methods to produce a new TCP function from the current set. (a) feature extraction selects a binary-valued function $k$ and TCP function $f$ from $F$. The new TCP function is defined as $k \circ f$, where $\circ$ represents function composition. For example, $k \circ f = g_{L2B} \circ \text{ip.dst}$ outputs the lower two bytes of the destination address. Conversely, (b) binary operation chooses two TCP functions $f, h \in F$. Then, we produce a new TCP function $\psi(f, g)$, where $\psi : \mathscr{F} \times \mathscr{F} \to \mathscr{F}$. If $f = \text{ip.seq}$, $h = \text{ip.dst}$, and $\psi$ is the bitwise XOR, the TCP function $\psi(f, g)$ returns the XOR result between the sequence number and destination address, denoted by $\text{ip.seq} \oplus \text{ip.dst}$.

There are a wide variety of ways to select a TCP function. Here, we prioritize simple TCP functions over complex ones when selecting TCP functions in (a) feature extractions and (b) binary operations. We assess the simplicity of a TCP function $f$ by the number of applied function compositions; below, $\tau_{\text{count}}(f)$ equals the minimum number of compositions plus one:

$$\tau_{\text{count}}(f) := \min\{i \mid f \in F_i\}, \text{ where} \quad (8)$$
$$F_i := \left\{k \circ f \mid k \in K, f \in F_{i'}\ (i' < i)\right\}$$

**Algorithm 1** Generate TCP Function

**Input :** $F \subseteq \mathscr{F}$ : *TCP functions*
$K$ : binary-valued functions
($k \in K$ implies $k : \mathscr{B} \to \mathscr{B}$)
$\Psi$ : binary operations on $\mathscr{F}$
($\psi \in \Psi$ implies $\psi : \mathscr{F} \times \mathscr{F} \to \mathscr{F}$)
**Output:** $f$ : a new *TCP function*

**Function** `select_TCP_function` $(F)$ **:**
  $f \leftarrow$ select $f \in F$ with probability $\frac{(1/\tau_{\text{count}}(f))^2}{\sum_{f \in F}(1/\tau_{\text{count}}(f))^2}$
  **return** $f$

**Function** `feature_extraction` $(F, K)$ **:**
  $f \leftarrow$ `select_TCP_function`(F)
  $k \leftarrow$ select $k \in K$ with a discrete uniform distribution
  **return** $k \circ f$

**Function** `binary_operation` $(F, \Psi)$ **:**
  $f \leftarrow$ `select_TCP_function`$(F)$
  $g \leftarrow$ `select_TCP_function`$(F)$
  $\psi \leftarrow$ select $\psi \in \Psi$ with a discrete uniform distribution
  **return** $\psi(f, g)$

**Function** `generate_TCPfunction` $(F)$ **:**
  $x \sim$ Uniform(0,1) `// random number from` [0, 1]
  **if** $x \leq 0.5$ **then**
    $f \leftarrow$ `feature_extraction`$(F, K)$
  **else**
    $f \leftarrow$ `binary_operation`$(F, \Psi)$
  **return** $f$

$$\cup \{\psi(f, g) \mid \psi \in \Psi, f \in F_{i_1}, g \in F_{i_2}\ (i_1, i_2 < i)\} \tag{9}$$

for $i = 2, 3, \cdots$. For instance,

$$\tau_{\text{count}}\left(g_{\text{L2B}}(\text{ip.seq}) \oplus g_{\text{L2B}}(\text{ip.dst})\right) \tag{10}$$
$$= \tau_{\text{count}}\left(g_{\text{L2B}}\left(\psi(\text{ip.seq}, \text{ip.dst})\right)\right) \tag{11}$$
$$= 3 \tag{12}$$

where $\psi$ denotes bitwise XOR. Finally, a TCP function $f \in F$ is selected with probability,

$$\frac{(1/\tau_{\text{count}}(f))^2}{\sum_{f \in F}(1/\tau_{\text{count}}(f))^2} \tag{13}$$

in (a) feature extractions and (b) binary operations.

`generate_TCPfunction` in Algorithm 1 is the pseudocode for generating a new TCP function. The pseudocode does not care about how the binary-valued function and $\psi : \mathscr{F} \times \mathscr{F} \to \mathscr{F}$ are selected; we can modify it according to the purpose.

### D. ANALOGY BETWEEN GENETIC ALGORITHM AND GENERATING TCP FUNCTIONS

The genetic algorithm (GA) is used in many research fields to produce high-quality solutions to optimization problems [61]. In the GA, a population of candidate solutions (individuals) iteratively evolves into better and better solutions via the use of biologically inspired operators, such as mutation, crossover, and selection. In each iteration, the fitness of every individual is evaluated, and the fit individuals are stochastically selected from the current population. Subsequently, these individuals are modified via recombination or random mutation.

Because we could not directly apply the GA to our problem, we leveraged the ideas behind it for generating TCP functions. Here, a TCP function corresponds to an individual, and the fitness of each TCP function is assessed via $\tau_{\text{count}}$. Specifically, we prefer choosing a TCP function $f$ with a small $\tau_{\text{count}}(f)$, which leads to simpler TCP functions. (a) feature extraction and (b) binary operations correspond to mutation and crossover. The output of these operations inherits the input features in the same way as mutation and crossover.

## REFERENCES

[1] S. Torabi, E. Bou-Harb, C. Assi, M. Galluscio, A. Boukhtouta, and M. Debbabi, "Inferring, characterizing, and investigating internet-scale malicious IoT device activities: A network telescope perspective," in *Proc. 48th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2018, pp. 562–573.

[2] D. D. Chen, M. Egele, M. Woo, and D. Brumley, "Towards automated dynamic analysis for linux-based embedded firmware," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2016, pp. 1–16.

[3] M. Antonakakis, T. April, and M. Bailey, "Understanding the Mirai botnet," in *Proc. USENIX Secur. Symp.*, 2017, pp. 1093–1110.

[4] B. Krebs. *KrebsOnSecurity Hit With Record DDoS*. Accessed: Dec. 5, 2016. [Online]. Available: https://krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos/

[5] L. Cashdollar. (2019). *Latest ECHOBOT: 26 Infection Vectors*. Accessed: Dec. 5, 2022. [Online]. Available: https://www.akamai.com/blog/security/latest-echobot-26-infection-vectors

[6] S. M. Bellovin, "Packets found on an internet," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 23, no. 3, pp. 26–31, Jul. 1993, doi: 10.1145/174194.174199.

[7] C. Fachkha and M. Debbabi, "Darknet as a source of cyber intelligence: Survey, taxonomy, and characterization," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1197–1227, 2nd Quart., 2016, doi: 10.1109/COMST.2015.2497690.

[8] P.-A. Vervier and Y. Shen, "Before toasters rise up: A view into the emerging IoT threat landscape," in *Research in Attacks, Intrusions, and Defenses*. Cham, Switzerland: Springer, 2018, pp. 556–576, doi: 10.1007/978-3-030-00470-5_26.

[9] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring internet denial-of-service activity," *ACM Trans. Comput. Syst.*, vol. 24, no. 2, pp. 115–139, May 2006, doi: 10.1145/1132026.1132027.

[10] N. Blenn, V. Ghiëtte, and C. Doerr, "Quantifying the spectrum of denial-of-service attacks through internet backscatter," in *Proc. 12th Int. Conf. Availability, Rel. Secur.*, Aug. 2017, pp. 1–10, doi: 10.1145/3098954.3098985.

[11] G. Gu, J. Zhang, and W. Lee, "BotSniffer: Detecting botnet command and control channels in network traffic," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2008, pp. 1–18.

[12] J. Mazel, R. Fontugne, and K. Fukuda, "Profiling internet scanners: Spatiotemporal structures and measurement ethics," in *Proc. Netw. Traffic Meas. Anal. Conf. (TMA)*, Jun. 2017, pp. 1–9, doi: 10.23919/TMA.2017.8002909.

[13] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*. Palo Alto, CA, USA: AAAI Press, 1996, pp. 226–231.

[14] T. Ban, M. Eto, S. Guo, D. Inoue, K. Nakao, and R. Huang, "A study on association rule mining of darknet big data," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–7, doi: 10.1109/ijcnn.2015.7280818.

[15] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 1993, pp. 1–10, doi: 10.1145/170035.170072.

[16] S. Lagraa, Y. Chen, and J. François, "Deep mining port scans from darknet," *Int. J. Netw. Manage.*, vol. 29, no. 3, p. e2065, May 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nem.2065

[17] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mechanics, Theory Exp.*, vol. 2008, no. 10, Oct. 2008, Art. no. P10008, doi: 10.1088/1742-5468/2008/10/p10008.

[18] F. Soro, M. Allegretta, M. Mellia, I. Drago, and L. M. Bertholdo, "Sensing the noise: Uncovering communities in darknet traffic," in *Proc. Medit. Commun. Comput. Netw. Conf. (MedComNet)*, Jun. 2020, pp. 1–8, doi: 10.1109/medcomnet49392.2020.9191555.

[19] D. Cohen, Y. Mirsky, M. Kamp, T. Martin, Y. Elovici, R. Puzis, and A. Shabtai, "DANTE: A framework for mining and monitoring darknet traffic," in *Computer Security—ESORICS 2020*. Cham, Switzerland: Springer, 2020, pp. 88–109, doi: 10.1007/978-3-030-58951-6_5.

[20] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, vol. 2, 2013, pp. 3111–3119.

[21] L. Gioacchini, L. Vassio, M. Mellia, I. Drago, Z. B. Houidi, and D. Rossi, "DarkVec: Automatic analysis of darknet traffic with word embeddings," in *Proc. 17th Int. Conf. Emerg. Netw. Exp. Technol.*, Dec. 2021, pp. 1-15, doi: 10.1145/3485983.3494863.

[22] M. Ring, A. Dallmann, D. Landes, and A. Hotho, "IP2Vec: Learning similarities between IP addresses," in *Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2017, pp. 657–666, doi: 10.1109/ICDMW.2017.93.

[23] E. Bou-Harb, M. Debbabi, and C. Assi, "On fingerprinting probing activities," *Comput. Secur.*, vol. 43, pp. 35–48, Jun. 2014, doi: 10.1016/j.cose.2014.02.005.

[24] C. K. Peng, S. V. Buldyrev, S. Havlin, M. Simons, H. Stanley, and A. L. Goldberger, "Mosaic organization of DNA nucleotides," *Phys. Rev. B, Condens. Matter*, vol. 49, no. 2, pp. 1685–1689, Feb. 1994, doi: 10.1103/physreve.49.1685.

[25] C. Han, J. Shimamura, T. Takahashi, D. Inoue, J. Takeuchi, and K. Nakao, "Real-time detection of global cyberthreat based on darknet by estimating anomalous synchronization using graphical lasso," *IEICE Trans. Inf. Syst.*, vol. E103.D, no. 10, pp. 2113–2124, Oct. 2020, doi: 10.1587/transinf.2020edp7076.

[26] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, Jul. 2008.

[27] C. Han, J. Takeuchi, T. Takahashi, and D. Inoue, "Automated detection of malware activities using nonnegative matrix factorization," in *Proc. IEEE 20th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Oct. 2021, pp. 548–556, doi: 10.1109/trustcom53373.2021.00085.

[28] D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. 13th Int. Conf. Neural Inf. Process. Syst.* Cambridge, MA, USA: MIT Press, 2000, pp. 535–541.

[29] H. Kanehara, Y. Murakami, J. Shimamura, T. Takahashi, D. Inoue, and N. Murata, "Real-time botnet detection using nonnegative tucker decomposition," in *Proc. 34th ACM/SIGAPP Symp. Appl. Comput.*, Apr. 2019, pp. 1337–1344, doi: 10.1145/3297280.3297415.

[30] Y.-D. Kim and S. Choi, "Nonnegative tucker decomposition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8, doi: 10.1109/CVPR.2007.383405.

[31] C. Fachkha, E. Bou-Harb, A. Keliris, N. Memon, and M. Ahamad, "Internet-scale probing of CPS: Inference, characterization and orchestration analysis," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2017, pp. 1–15, doi: 10.14722/ndss.2017.23149.

[32] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-26, no. 1, pp. 43–49, Feb. 1978, doi: 10.1109/TASSP.1978.1163055.

[33] J. Kornblum, "Identifying almost identical files using context triggered piecewise hashing," *Digit. Invest.*, vol. 3, pp. 91–97, Sep. 2006, doi: 10.1016/j.diin.2006.06.015.

[34] C. Gates, "Coordinated scan detection," in *Proc. NDSS*, vol. 9, 2009, pp. 1–13.

[35] E. Bou-Harb, M. Debbabi, and C. Assi, "A time series approach for inferring orchestrated probing campaigns by analyzing darknet traffic," in *Proc. 10th Int. Conf. Availability, Rel. Secur.*, Aug. 2015, pp. 180–185, doi: 10.1109/ares.2015.9.

[36] C. M. Library, *Trigonometric Series*. Cambridge, U.K.: Cambridge Univ. Press, 2002.

[37] G. Welch and G. Bishop, "An introduction to the Kalman filter," in *Proc. SIGGRAPH*, 1995, pp. 1–16.

[38] S. Torabi, E. Bou-Harb, C. Assi, E. B. Karbab, A. Boukhtouta, and M. Debbabi, "Inferring and investigating IoT-generated scanning campaigns targeting a large network telescope," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 1, pp. 402–418, Jan. 2022, doi: 10.1109/tdsc.2020.2979183.

[39] M. Safaei Pour, A. Mangino, K. Friday, M. Rathbun, E. Bou-Harb, F. Iqbal, S. Samtani, J. Crichigno, and N. Ghani, "On data-driven curation, learning, and analysis for inferring evolving Internet-of-Things (IoT) botnets in the wild," *Comput. Secur.*, vol. 91, Apr. 2020, Art. no. 101707, doi: 10.1016/j.cose.2019.101707.

[40] J. Jung, V. Paxson, A. W. Berger, and H. Balakrishnan, "Fast portscan detection using sequential hypothesis testing," in *Proc. IEEE Symp. Secur. Privacy*, May 2004, pp. 211–225, doi: 10.1109/SECPRI.2004.1301325.

[41] Z. Durumeric, M. Bailey, and J. A. Halderman, "An internet-wide view of internet-wide scanning," in *Proc. 23rd USENIX Conf. Secur. Symp.*, 2014, pp. 65–78.

[42] F. Shaikh, E. Bou-Harb, N. Neshenko, A. P. Wright, and N. Ghani, "Internet of Malicious things: Correlating active and passive measurements for inferring and characterizing internet-scale unsolicited IoT devices," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 170–177, Sep. 2018, doi: 10.1109/MCOM.2018.1700685.

[43] H. Griffioen and C. Doerr, "Discovering collaboration: Unveiling slow, distributed scanners based on common header field patterns," in *Proc. IEEE/IFIP Netw. Operations Manage. Symp. (NOMS)*, Apr. 2020, pp. 1–9, doi: 10.1109/noms47738.2020.9110444.

[44] J. Xie, B. K. Szymanski, and X. Liu, "SLPA: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process," in *Proc. 11th Int. Conf. Data Mining Workshops*, Dec. 2011, pp. 344–349, doi: 10.1109/icdmw.2011.154.

[45] M. Lastovicka, T. Jirsik, P. Celeda, S. Spacek, and D. Filakovsky, "Passive Os fingerprinting methods in the jungle of wireless networks," in *Proc. IEEE/IFIP Netw. Operations Manage. Symp. (NOMS)*, Apr. 2018, pp. 1–9, doi: 10.1109/noms.2018.8406262.

[46] A. Tanaka, C. Han, T. Takahashi, and K. Fujisawa, "Internet-wide scanner fingerprint identifier based on TCP/IP header," in *Proc. 6th Int. Conf. Fog Mobile Edge Comput. (FMEC)*, Dec. 2021, pp. 1–6, doi: 10.1109/fmec54266.2021.9732414.

[47] S. Herwig, K. Harvey, G. Hughey, R. Roberts, and D. Levin, "Measurement and analysis of Hajime, a peer-to-peer IoT botnet," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2019, pp. 1–15, doi: 10.14722/ndss.2019.23488.

[48] R. Graham. *MASSCAN: Mass IP Port Scanner*. Accessed: Dec. 7, 2022. [Online]. Available: https://github.com/robertdavidgraham/masscan

[49] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast internet-wide scanning and its security applications," in *Proc. 22nd USENIX Secur. Symp.*, 2013, pp. 605–620.

[50] C. Han, J. Takeuchi, T. Takahashi, and D. Inoue, "Dark-TRACER: Early detection framework for malware activity based on anomalous spatiotemporal patterns," *IEEE Access*, vol. 10, pp. 13038–13058, 2022, doi: 10.1109/ACCESS.2022.3145966.

[51] R. Daws. *IBM X-Force Discovers Mozi Botnet Accounts for 90% of IoT Traffic*. Accessed: Nov. 15, 2020. [Online]. Available: https://www.iottechnews.com/news/2020/sep/18/ibm-xforce-mozi-botnet-iot-traffic/

[52] Microsoft Security Response Center. (2021). *Clarified Guidance for CVE-2021-34527 Windows Print Spooler Vulnerability*. Accessed: Jan. 16, 2023. [Online]. Available: https://msrc-blog.microsoft.com/2021/07/08/clarified-guidance-for-cve-2021-34527-windows-print-spooler-vulnerability/
[53] R. Tyagi, T. Paul, B. S. Manoj, and B. Thanudas, "Packet inspection for unauthorized OS detection in enterprises," *IEEE Secur. Privacy*, vol. 13, no. 4, pp. 60–65, Jul. 2015, doi: 10.1109/msp.2015.86.
[54] P. Matousek, O. Rysavý, M. Grégr, and M. Vymlátil, "Towards identification of operating systems from the internet traffic–IPFIX monitoring with fingerprinting and clustering," in *Proc. 5th Int. Conf. Data Commun. Netw.*, 2014, pp. 1–7.
[55] SANS Institute. (2003). *Passive OS Fingerprinting Update*. Accessed: Jan. 18, 2023. [Online]. Available: https://www.dshield.org/diary/Passive+OS+Fingerprinting+Update/18
[56] RootKiter. (2018). *HNS Botnet Recent Activities*. Accessed: Dec. 5, 2022. [Online]. Available: https://blog.netlab.360.com/hns-botnet-recent-activities-en/
[57] S. Srinivasa, J. M. Pedersen, and E. Vasilomanolakis, "Open for hire: Attack trends and misconfiguration pitfalls of IoT devices," in *Proc. 21st ACM Internet Meas. Conf.*, Nov. 2021, pp. 195–215, doi: 10.1145/3487552.3487833.
[58] P. Richter and A. Berger, "Scanning the scanners: Sensing the Internet from a massively distributed network telescope," in *Proc. Internet Meas. Conf.*, Oct. 2019, pp. 144–157, doi: 10.1145/3355369.3355595.
[59] M. S. Pour, J. Khoury, and E. Bou-Harb, "HoneyComb: A darknet-centric proactive deception technique for curating IoT malware forensic artifacts," in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp. (NOMS)*, Apr. 2022, pp. 1–9, doi: 10.1109/noms54207.2022.9789827.
[60] T. Takahashi, Y. Umemura, C. Han, T. Ban, K. Furumoto, O. Nakamura, K. Yoshioka, J. Takeuchi, N. Murata, and Y. Shiraishi, "Designing comprehensive cyber threat analysis platform: Can we orchestrate analysis engines?" in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops Other Affiliated Events (PerCom Workshops)*, Mar. 2021, pp. 376–379, doi: 10.1109/PERCOMWORKSHOPS51409.2021.9431125.
[61] Z. Wang and A. Sobey, "A comparative review between genetic algorithm use in composite optimisation and the state-of-the-art in evolutionary computation," *Composite Struct.*, vol. 233, Feb. 2020, Art. no. 111739, doi: 10.1016/j.compstruct.2019.111739.

**AKIRA TANAKA** received the degree in mathematics from Kyushu University, in 2022. He is currently a Researcher with the National Institute of Information and Communications Technology (NICT), Japan. From 2022 to 2023, he has been a Visiting Researcher at Kyushu University, Japan. His research interest includes analyzing and solving cybersecurity problems, especially darknet traffic, using machine learning.

**CHANSU HAN** received the B.E. degree in computer science and the M.S. and Ph.D. degrees in informatics engineering from Kyushu University, in 2016, 2018, and 2021, respectively. He is currently a Researcher with the National Institute of Information and Communications Technology (NICT), Japan. His research interest includes analyzing and solving problems in the cybersecurity field (especially networks and malware) using machine learning.

**TAKESHI TAKAHASHI** (Member, IEEE) received the Ph.D. degree in telecommunications from Waseda University, in 2005. He worked at the Tampere University of Technology from 2002 to 2004 as a Researcher, Waseda University as a JSPS Research Fellow from 2004 to 2006, and Roland Berger Ltd. from 2006 to 2009 as a Business Consultant. Since 2009, he has been with the National Institute of Information and Communications Technology, where he is currently a Research Manager. He was a Visiting Research Scholar at the University of California, Santa Barbara, from 2019 to 2020. His research interests include cybersecurity and machine learning.