

Received April 5, 2022, accepted April 26, 2022, date of publication April 29, 2022, date of current version May 12, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3171226

A Comprehensive Analysis of Today's Malware and Its Distribution Network: Common Adversary Strategies and Implications

SIWON HUH¹, SEONGHWAN CHO², JINHO CHOI², SEUNGWON SHIN³, (Member, IEEE), AND HOJOON LEE¹

¹Department of Computer Science and Engineering, Sungkyunkwan University, Jangan-gu, Suwon-si 16419, Republic of Korea

²Graduate School of Information Security, School of Computing, Korea Advanced Institute of Science and Technology (KAIST), Yuseong-gu, Daejeon 34141, South Korea

³School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Yuseong-gu, Daejeon 34141, South Korea

Corresponding author: Hojoon Lee (hojoon.lee@skku.edu)

This work was supported by National Research Foundation of Korea (NRF) Grant by the Korean Government through (NRF-2022R1C1C1010494), and Ministry of Science, ICT (MSIT), Korea, under the High-Potential Individuals Global Training Program (IITP-2021-0-02104).

ABSTRACT Malware has plagued the internet and computing systems for decades. The war against malware has always been an arms race. Researchers and industry have constantly improved detection and prevention methodologies against increasingly more evasive malware. Keeping up with the constantly changing adversary tactics for evading defensive efforts and maintaining an efficient malware supply chain is imperative to stay ahead in the competition. In this paper, we present a large-scale and comprehensive analysis of the current state of malware distribution. For the analysis, we accumulated a dataset that contains 99,312 malware binary samples from 38,659 malware distribution sites over 287 days. Using our dataset, we perform a comprehensive analysis of the collected malware binaries and URLs to provide up-to-date statistics and insights into the adversary strategies. We analyze both malware distribution sites and malware binaries collected from them. Regarding binary analysis, we perform a multifaceted analysis on the characteristics on the collected binaries, including malware family label classification and file similarity-based clustering. With distribution site analysis, we analyze the IP addresses, domains, AS registration distribution and URL lexical distribution of malware distribution sites. We further discuss the statistical relationship between malware families and their distribution domains. Most importantly, we discuss the current trends in malware distribution today and reveal adversary strategies through our extensive amount of analysis results. Then, we suggest future directions for fight against malware distribution.

INDEX TERMS Malware distribution network, cyber security, malware classification, malware features.

I. INTRODUCTION

For decades, the war against malware has been an arms race. Malware distribution today is highly industrialized, and adversaries actively employ various evasion techniques at a large scale to avoid malware detection schemes. Due to their evasive behavior and rapid evolution, understanding malware and its distribution network have always been the topic of interest for researchers. Several works have conducted focused studies to understand a specific

The associate editor coordinating the review of this manuscript and approving it for publication was S. K. Hafizul Islam¹.

issue or phenomenon in malware distribution. For instance, Caballero *et al.* [1] studied the malware distribution in the Pay-Per-Install (PPI) ecosystem. Thomas *et al.* [2] focused on the Potentially Unwanted Program (PUP) distributed by PPI services. Furthermore, Kotzias *et al.* [3] and Ife *et al.* [4] conducted a large-scale analysis on identifying the distribution patterns of PUPs with the metadata collection provided by AV vendors. Kwon *et al.* [5] presented a large-scale analysis on Windows executables droppers, which is conducted with the metadata collection provided by an AV vendor. Ugarte-Pedrero *et al.* [6] provided a dedicated analysis on a large number of malware samples (i.e., 172K samples),

focusing on Windows executables collected within 24 hours to analyze the effectiveness of AV vendor filtering schemes on malware samples. Lastly, Alrawi *et al.* [7] studied 166,772 IoT malware samples to identify the difference in the evasion techniques of IoT malware from the traditional malware.

Some works have also studied the malware measurement to understand the distribution of malware or distribution sites on a large scale. Hu *et al.* [8] performed a large-scale malware clustering based on static features with 132,234 malware binaries. Tanaka *et al.* [9] conducted a long-term analysis on malware distribution sites, with their 43,304 URLs collected in 19 months.

Besides, there also have been approaches on enhancing the efficiency and accuracy of automated malware detection, using clustering [8], [10]–[12] and deep learning [13]–[18]. Recent works have applied the latest clustering and deep learning techniques for malware detection. Notably, Chakraborty *et al.* [12] proposed a combined approach of clustering and supervised classifiers that deals with classifying unknown malware samples into families. Ferrag *et al.* [19] applied federated deep learning to handle diversifying features of malware in IoT infrastructure. Some studies have also worked on developing an appropriate dataset for the fair measurement of machine learning-based malware detection [20]–[23].

Our work provides a *comprehensive* measurement of the *current* state of the malware distribution and reveals the latest adversary strategies in the process. Our work is comprehensive in the following two aspects: First, we do not limit our target to a specific class of malware or a particular phenomenon. Rather, we consider all malware types that users frequently encounter through using public data sources. This differs our work from previous works that have only considered the malware binaries of certain behavior [7], [24], or a specific subset of malware binaries [1]–[3], [5]. Our work, on the other hand, captures a more comprehensive view on the state of malware distribution and adversary strategies. Second, we analyze *both* malware distribution sites and the binaries collected from the URLs. There are also existing works [3], [4] that have also conducted analysis on both malware binaries and URLs, but we point out that their dataset is rather dated, which is collected no later than 2016. By analyzing both binaries and their distribution sites, we take a closer look into the *current* adversary strategies from malware binary generation to their distribution.

In this work, we collect *live* malware binaries and their distribution URLs from public data sources in real-time with the crawler we implement. Then, we identify the malware family distribution and the similarities among malware families. For some peculiar cases, we provide in-depth case studies focusing on their evasion techniques through manual analysis. We then propose a clustering approach based on file similarity to provide better heuristics on classifying malware families variants and also to provide more accurate classification than vendor-given labels. We also try to identify the mutation strategy of current malware by comparing

binary differences of the files in the same cluster. With the URL collection, we statistically analyze the distribution of IP, domain, AS registration, and geographical location to identify the distribution trend of malware. Finally, we provide discussions with implications from our findings and suggest future directions for the research community and industry in cyber security.

The remainder of this paper is organized as follows: § II explains the related works and provides a dedicated comparison that distinguishes our work from the existing works. We explain our data collection methodology in § III, and provide an in-depth analysis on malware binaries in § IV. In § V, we discuss the result of file similarity based clustering. Then, we present the analysis on the statistics of malware distribution site URLs in § VI. § VII provides discussions and future directions for the research community, based on the results from our analysis. Lastly, concluding remarks are drawn in § VIII.

In all, we summarize our contributions as the following:

- We design a malicious binary and URL collection system that collects *live* malware binaries from newly reported malware distribution URLs. Using the system, we build a dataset over 287 days that contains 99,312 malware binary samples from 38,659 malware distribution site URLs (§ III).
- We provide an in-depth analysis of malware binaries in the dataset to illustrate the current trends in the malware distribution industry and the detection evasion strategies of attackers (§ IV).
- We discuss the trends in distribution sites through the statistics from our dataset (§ VI). More specifically, we perform a clustering method on the collected malware binaries to classify malware based on their binary similarity to complement the current malware classification convention that is different from vendor to vendor (§ V).

II. RELATED WORK

Table 1 shows the comparison of our work to the existing works. Our work conducts a study on the distributed malware binaries and malware distribution networks to capture the latest trends and adversary strategies in malware distribution.

Some existing works focused on understanding specific phenomena. For instance, existing works have sought to understand *Potentially Unwanted Programs (PUPs)* in depth [2], [3]. Some works focused on the behavior and characteristics of malware binaries themselves [7], [24], and a few works are even more focused, only considering Windows executables [5], [6], or distribution URLs [9]. On the other hand, some other works focused on identifying the feature of malware binary or their distribution sites. [8] worked on clustering malware based on its static feature, and [9] provided a long-term analysis on malware distribution URLs. Otherwise, our work covers all types of malware and both

TABLE 1. Comparison of literature on malware analysis. Our work provides a comprehensive analysis overview on the state of malware distribution today through both malware binary and distribution site analysis.

Work	Year	Data Source	Analyzed Data	Collection Period	Data Size	Analysis Objective
[1]	2011	Self-collected	Binaries (From PPI services)	6 months	9,153	Understanding pay-per-install economy of malware
[8]	2013	Public dataset	Binaries	Unknown	132,234	Static feature based malware clustering
[5]	2015	AV vendor dataset (metadata only)	Binaries (PE only)	Unknown	24M binaries	Analyzing distribution pattern of malware downloaders
[3]	2016	AV vendor dataset (metadata only)	Binaries / URLs (PUP only)	18 months	2.6M binaries 290K URLs	Understanding difference between malware and PUP distribution
[2]	2016	Self-collected	Binaries (PUP only)	12 months	1,211	Understanding PUP distribution of commercial services
[9]	2017	Self-collected	URLs	19 months	43,034	Long-term analysis on malware distribution sites
[4]	2019	AV vendor dataset (metadata only)	Binaries / URLs	12 months	21M binaries 12M URLs	Comparing distribution pattern between malware and PUP
[6]	2019	AV vendor dataset	Binaries (PE only)	24 hours	172,612	Understanding challenges of filtering new malware in vendor perspective
[7]	2021	Self-collected	Binaries (IoT malware only)	8 months	166,772	Understanding life cycle of IoT malware
This work	2021	Self-collected	Binaries / URLs	9.4 months	99,312 binaries 38,659 URLs	<ul style="list-style-type: none"> • Comprehensive malware binary characterizing with statistics • In-depth reverse engineering on previously unknown binary • Similarity-based binary clustering • Malware distribution site statistics • Analysis on current adversary strategies based on findings

their distribution URLs to figure out the comprehensive trend of malware distribution.

Malware distribution and defensive efforts are constantly engaging in an endless arms race, and adversary strategies change. For this reason, an up-to-date analysis on malware is of value to researchers and industry alike. To the best of our knowledge, [7] is the most recent work on large-scale malware analysis. However, this work limits its target to Linux-based IoT malware. The work of Ife *et al.* [4] also covers comprehensive data, including malware binaries and URLs, but we point out that the dataset used in this work is rather dated (2016). We also differ our work from other works [3]–[5] that leveraged the AV vendor dataset which only consists of metadata, not the real binaries. We emphasize that our work shows the current state of malware distribution and attempts to maintain representativeness by crawling real-time data.

A. PUP DISTRIBUTION ECONOMY

Several works focused on the analysis of PUPs [1]–[3]. PUPs are a subset of malware and also distributed in a large-scale. However, PUPs differ from malware in that they are often not outright malicious, but still exhibit a certain degree of intrusive behavior (e.g., browser toolbars and adwares). Notably, Caballero *et al.* [1] reverse-engineered the software bundle and implemented the “milker”, or data collection, tool, which periodically downloads the target program that pay-per-install installers try to download. They found out that some families repack their target program more than twice a day. Thomas *et al.* [2] also developed a tool that collects PUPs and the internal operations of their distributors. They

examined deceptive behaviors of the PUP distributors that attempt to evade user protections. These works are similar to our work in that both collected multiple samples that were updated on the server-side. However, our work is more general because we consider all types of malware, not only the PUP.

B. LARGE-SCALE MALWARE MEASUREMENTS

Recently, researchers studied a large malware corpus from various data sources, such as online sandbox analysis systems and commercial sandbox systems from Anti-Virus (AV) vendors. Ugarte-Pedrero *et al.* [6] dissected the whole dataset from a single AV vendor collected for 24 hours. They studied how much effort it takes to process malware we face daily. They filtered out most of the samples using VirusTotal and sandbox analysis and showed that only large companies could process the emerging malware samples, as abundant computation power and human resources are needed. This work is somewhat similar to our work. However, most of their data consist of polymorphic malware families, so their results can be biased towards those families. Kwon *et al.* [5] constructed a downloader graph based on IPS and AV telemetry data and showed that there are differences between benign and malicious malware distribution patterns. However, they used Symantec antivirus and IPS telemetry as the ground dataset. We point out that the data from a vendor can also be biased toward polymorphic families, and HTTPS-based distribution is missing since IPS data only considered PE files sent over HTTP. Notably, Ife *et al.* [4] analyzed 129 million download events consisting of 21 million unique binaries and 12 million URLs that had occurred in over one year. However, the dataset

from Symantec is too old (2015 - 2016) and has only meta information, not real files that could be checked by VirusTotal. This research did not satisfy our demand to analyze the binary file in the real world. Lastly, Ugarte-Pedrero *et al.* [6] collected the samples from a large security company over a period of 24 hours. Their objective was to understand how “fresh” these files were by using VirusTotal, and 90% of their collected samples were also present in the VirusTotal. The paper focused on classifying every single Windows binary collected in a single day.

There are also previous works on collecting malware samples and URLs. Singhal and Levine [24] counter-plot the drive-by-download malware by visiting malicious URLs. They designed an own crawler that collects malware binaries linked to the URLs uploaded in public malware feeds, which is similar to our work. However, they worked with a small size of data, which was 1,414 unique malware binaries. Tanaka *et al.* [9] investigated the malware download sites focusing on the time-series variation of malware. The authors only focus on the change of malware hashes and disregard the difference of malware binary itself.

Hu *et al.* [8] performed a large-scale (over 130,000 samples) malware clustering. Their system, called *MutantX-S*, achieves high-speed binary sample clustering through the use of machine learning-based hashing techniques. In terms of accuracy, their evaluation showed their system could predict the correct label of a new unknown malware sample with 80% accuracy.

In addition, there has been a work on large-scale analysis of the emerging IoT malware ecosystem. Alrawi *et al.* [7] presented measurement of over 166K Linux-based IoT malware samples and tried to figure out the differences of IoT malware from the traditional malware. Their result shows that the evasion techniques of existing malware such as polymorphism and anti-analysis are already incorporated in IoT malware, and the behavior of IoT malware is still similar to the traditional malware. However, they point out that the flooding number of new IoT devices might embed the unrealized potential of IoT malware development.

C. DATA ANALYTICS FOR SECURITY

We expect machine learning can be adapted to advance automated malware analysis's efficiency and accuracy. A number of deep learning-based cyber-attack detection mechanisms have already been proposed [13]–[18]. Notably, many public cyber intrusion datasets have been accumulated for research purposes [20], [25]–[28]. Against such datasets, deep learning-based approaches have shown great performance with low false positive [29]–[32]. Recently, defense mechanisms that adopt federated deep learning have been proposed in academia to handle diversifying threats and large attack surfaces of IoT infrastructure [33]–[35]. However, the comprehensive analysis on federated learning-based intrusion detection of Ferrag *et al.* [19] has shown that they can even outperform the centralized deep learning-based approaches.

III. DATA COLLECTION METHOD

In this section, we explain our data collection methodology. We explain our choice of data sources, our crawler's design, and a summary of the collected data. We collected 38,659 malware distribution sites (i.e., malware URLs) and 99,312 malware samples in 287 days of the data collection period, from 09-19-2019 to 07-01-2020. We accumulated a representative dataset for malware distributed today. Thus, we chose our data source as public malware feeds where malicious or suspicious URLs encountered by users are reported. Our binaries are collected from such URLs while they are in *active distribution*. This way, our dataset contains the URLs and binaries from the malware distribution campaigns that were active during our collection period. In other words, all data in our dataset are the URLs and binaries that users come across on the internet. We built a crawler that is dedicated to our data collection method; it collects the distribution sites and gathers linked malware binaries in real-time.

A. DATA COLLECTION SOURCE SELECTION

We chose multiple public sources, such as URLHaus [36] and VxVault [37], as our data source for malware distribution URLs and also collected the linked malware binaries. These websites provide up-to-date malware binary and distribution site URLs, which are reported by users. Also, they perform crosschecks against malware analysis platforms such as VirusTotal [38], and this allows us to trust the URLs confirmed to be malicious. In addition to downloading malware in the malware distribution URLs in the sources mentioned above, we collect malware binaries from HybridAnalysis [39] and Joe Sandbox [40]. These services are more geared towards malware binary analysis. We also collect malware binaries from commercial dynamic malware analysis engines such as Anyrun [41]. Note that these sources are also run by user reports, which aligns with the rationale behind collecting distribution URLs from public sources. We consider all the data reported by users suspicious, so we also include the binaries and URLs that are decided as “potentially malicious” or “unknown” in our data collection. We found that malware is included in this data, and we discuss this in § V.

B. DATA COLLECTION SYSTEM OVERVIEW

Figure 1 illustrates the overview of the data collection system that we implemented. The system crawls the data sources explained above periodically to collect two types of data: malware distribution URLs and malware binaries uniquely identifiable through the MD5 hashes. For all reported malware distribution URLs, we also collect the malware binaries being distributed from the URL. That is, we are collecting malware binaries from the URLs in addition to the malware binaries report sources (e.g., Anyrun, Joe Sandbox). The binaries collected from the URLs may or may not overlap the binaries from the malware binary sources. This means that

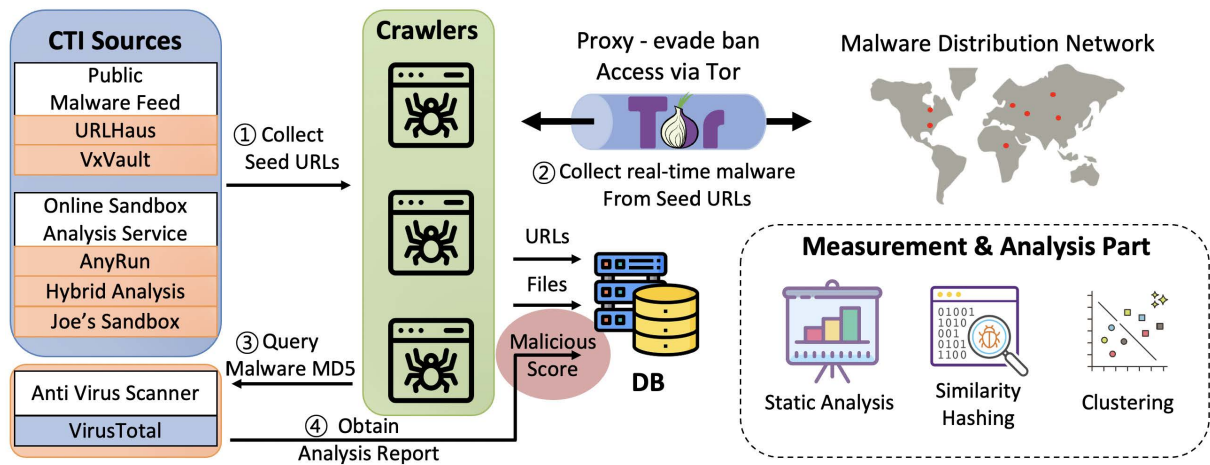


FIGURE 1. Data Collection Overview. ① Crawler collects reported URLs from public malware feeds and malware binary analysis services. ② To collect the malware binary linked to the collected URLs, crawler accesses the malware URLs in real-time, with Tor avoid blacklisting by the adversaries. ③ Crawler queries the MD5 hashes of collected malware binaries to VirusTotal to retrieve AV vendor reports the given binary. ④ Timestamped distribution site URL, binary hash, binary file, VirusTotal results are saved in the database.

our system is consolidating the data from multiple sources to capture the actively distributed malware binaries at the time.

We optimized the crawling period for each website based on our heuristics on the website's update frequency (10 min ~ 1 hour). Keeping up with the newly reported links is crucial since the distribution URLs have finite and rather unpredictable uptime. This is because the URL may be taken down by the authorities and also subject to frequent malware binary replacement for evasion of signature-based blacklisting (e.g., `https://malware.site/mw-v1` → `https://malware.site/mw-v2`). Hence, periodic collection is necessary in order to obtain the malware being distributed from the URL at that exact time.

C. DATA COLLECTION STRATEGIES

Our system connects through Tor [42] when collecting the malware binaries from the reported distribution URLs. We observe that the malware distribution sites blacklist frequent visitors that are suspected to sabotage malware data collection systems such as our crawler. Another strategy we developed during the early testing phase of our collection system was to check for directory listings of the malware distribution sites. We discovered that the malware distributors often prepare malware binaries on the distribution site web server before they start distributing them. These binaries were sometimes previously unknown, suggesting that they might be malware binaries on pre-distribution. As a result, this aggressive crawling strategy enabled us to compile 220% more samples from the existing data.

D. CHALLENGES IN COLLECTING LIVE MALWARE DATASET

We faced formidable challenges in collecting a *live* dataset that consists of distribution site URLs and the distributed binaries.

First, the malware distributions tend to be very dynamic. We observed that the malware distributors deliberately

change distribution site URLs as well as the distributed binaries (e.g., `xxx.com/aa/bb.exe` → `xxx.com/ccc/ddd.exe`). Once the distributors realize that AV vendors have blacklisted the distribution site, they quickly abandon the URL and switch to a new one. A single distribution site may suddenly swap the distributed malware binary as the distributors discover that the binary has been blacklisted. This forced our data collection system to be real-time to keep up with the constantly changing distribution sites.

Second, during our data collection period, we found that the distributors ban the IP addresses of clients that visit repeatedly (e.g., our data collection system). Hence, we resorted to using Tor to avoid being blacklisted. Due to the added network delay from Tor, we had to invest more resources into our data collection system to keep up with the malicious URL reports piling up. As such, our data collection system was required to be real-time to keep up with the constantly changing distribution sites.

Third, during our data collection period, we experienced that the distributors ban the IP addresses of clients that visit repeatedly (e.g., our data collection system). Hence, we resorted to using Tor to avoid being blacklisted. Due to the added network delay from Tor, we had to invest more resources into our data collection system to keep up with the malicious URL reports piling up. Overall, maintaining the above efforts for 287 days without a single downtime had been a formidable challenge.

E. VALUE OF OUR DATASET

Our dataset provides several advantages in performing a comprehensive measurement on the current state of malware distribution. Our dataset includes both malware binaries *linked* with their distribution sites. This linkage between the distribution site and the malware binaries makes our dataset valuable since it allows us to study the adversary strategy in preparing binaries and managing the distribution sites.

TABLE 2. Most frequently observed file formats in malware payload.

Content Type	File Type
openxmlformats, dosexec, ms-powerpoint, x-executable, msword	exe, dll, doc, ppt, pptx, docx, hwp, bat, ps1

Therefore, we differentiate our dataset from those that only contain malware binaries. Also, as we collect malware distribution sites and distributed binaries from user report-based public sources, our data collection represents the real-world malware that users encounter on the Internet. This also differentiates our dataset from AV-vendor-provided ones, which might be biased. Lastly, trends in malware binaries and distribution sites change rapidly, and thus, a dataset loses its value over time. The fact that the most recent comprehensive dataset (e.g., binaries + URLs) was collected in 2016 [4] motivated this work.

IV. MALWARE BINARY ANALYSIS

In this section, we provide an in-depth analysis of the collected malware binary samples to identify the behaviors and characteristics of malware distributed today. We perform a multifaceted analysis on the characteristics such as file type, family, and obfuscation mechanism to illustrate the current state of malware distribution. Also, we introduce our method for quantifying the similarities among different malware binaries. One observation we make through the process is that malware family classification reported by the AV companies shows large variances. That is, the classification often does not reflect the actual similarities of the binaries, but is rather determined by the vendor-specific classification methods. We report our analysis on our malware binary samples and discuss the implications of our analysis.

We use VirusTotal [38] reports for the identification and classification of the collected file samples. VirusTotal provides malware detection test results that consolidate the reports from more than 70 AV vendors. The report contains the decisions of vendors whether a given binary sample is malicious or benign, while the binary hash value may not have a record in the vendor's database (i.e., unknown). Additionally, VirusTotal offers malware classification and malware family analysis.

We classify the collected binary samples in our dataset into {malicious, potentially malicious, benign, and unknown}. The results from the multiple AV vendors are often not in consensus. Therefore, we must establish a classification method for our binary samples. We use a threshold of 30% for classifying a sample as *malicious*. That is, a sample is classified as malicious if more than 30% of the reports from multiple AV vendors collected from VirusTotal claim that the sample is *malicious* ($r_{mal} > 30\%$). We classify the samples that has lower than 30% but at least one AV vendor report that claims it malicious as *potentially malicious* ($0 < r_{mal} < 30\%$). Benign samples have no AV vendor report that claims that it is malicious ($r_{mal} = 0\%$), and *unknown* samples are the ones with no report ($\# r_{mal}$). Establishing a ground truth for the

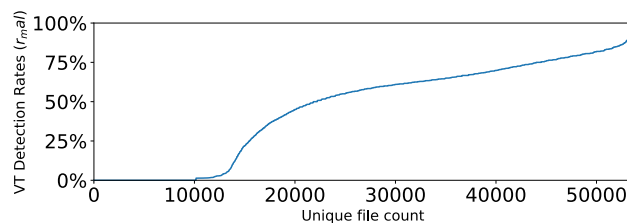


FIGURE 2. Cumulative Distribution of VirusTotal Detection Rates.

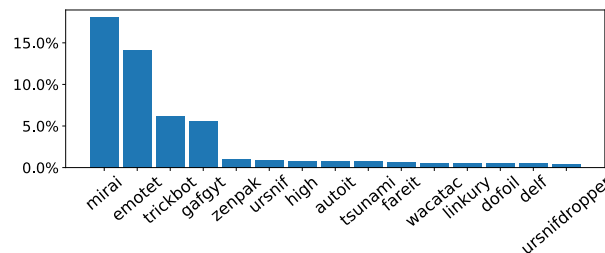


FIGURE 3. Top 15 Frequent AVClass Family.

binary samples is a daunting challenge for all analytic works that collect malware samples from the wild. For this reason, we use multi-vendor reports from VirusTotal as a source of ground truth and use a conservative threshold (30%) as in previous works [5], [43].

The distribution of binary sample classification is presented in Figure 2. 53,520 samples among the 99,312 had at least 10 AV vendor reports while 45,792 samples had no reports hence classified as unknown. Among the 53,520 samples, 37,317 file samples (69.7%) are classified as malicious, 4,235 samples (7.9%) as potentially malicious, and 11,968 samples (22.4%) as benign. As a result, we base our malware binary analysis on the 37,317 samples that are highly likely to be malicious.

A. MALWARE FAMILY DISTRIBUTION

We measure the distribution of malware family of the binary samples in our dataset. With this particular analysis, we exclude the unknown and potentially malicious samples. That is, we only consider the samples that are highly likely to be malicious ($r_{mal} > 30\%$) in order to derive a reliable and representative statistics on the malware families actively distributed today.

1) MALWARE FAMILY IDENTIFICATION FOR BINARY SAMPLES

We draw the malware family distribution from the 37,317 binary samples classified as malicious to determine the most commonly distributed malware today. We utilize the AVCLASS tool [44] for grouping the collected malware binaries into malware families, as previous research on malware clustering did [45]–[48]. Existing AV vendors also provide family names or classes on the malware. However, we found that vendors usually give different names to a malware binary. We surmise that this is due to the different naming conventions and internal malware analysis processes. AVCLASS

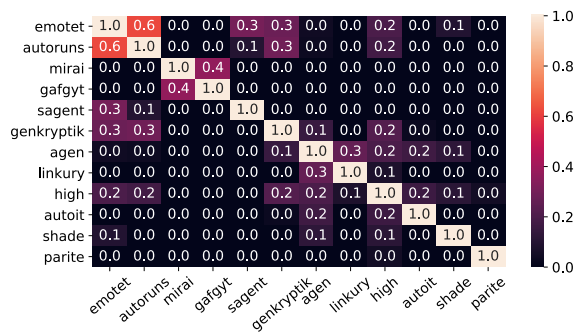


FIGURE 4. The heatmap of similarity measure between malware families.

is an open-source automatic malware labeling tool that normalizes multiple malware tags from multiple public malware analysis services for a malware sample. Therefore, we concluded that using AVCLASS allows us to better group malware samples with similar behavior and characteristics together instead of using the family classification from a single AV vendor.

Malware binary classification labels from AV vendors are often in disagreement, calling for a standardized and more descriptive conventions for malware family classification.

Figure 3 shows the top 15 family distributions of malware samples according to the aforementioned AVCLASS malware family aggregation. As a result, we found 6,267 families from our malware samples, suggesting a diverse malware family. Additionally, we discovered 1,070 singleton families (2.8% of total malware samples) that contain only a single malware sample. Notably, the top 4 families, Mirai, Emotet, Trickbot, and Gafgyt families make up more than 43% of all malware samples. The dominance of large malware families and the small proportion of singleton families indicate that most of the malware is not developed from scratch but is likely to be produced by mutilating existing malware.

Mirai, Emotet, Trickbot, and Gafgyt were the most commonly distributed malware today.

2) SIMILARITIES AMONG MALWARE FAMILIES

We measure the similarities among the malware families and create a two-dimension matrix with labels and samples, as shown in Figure 4. We normalize the detection result from VirusTotal using AVCLASS2. When given a hash value as an input, AVCLASS2 produces the name and number of each class that is flagged by AV engines. Using AVCLASS2, we again query the hashes we once labeled with AVCLASS, which only produces one representative label for each hash. We query the hashes for each label and accumulate the number produced by AVCLASS2 for each other labels. Then, each label will generate a vector, representing the number of other labels that can be produced by various AV engines. We treat this as a probability vector for each label and try to evaluate the distance among those vectors to compute similarity among the labels. Since the result sample vectors

to sample the comparison result are too small to show the similarity, we employ the square root of cosine similarity between vectors of two families.

In Figure 4, we observe the high similarity between some families. For example, the Emotet family shows an exceptionally high similarity with autoruns (0.6), and also shows some similarities with sagent, genkryptik, and high malware families. The Mirai family shows similarity to Gafgyt family. We found that the current labeling system provides no insight into the characteristics of the malware families, not to mention the conflicting labels reported by the AV vendors. We expect that i) cooperation among the AV vendors to standardize the malware family labels and ii) labeling convention that reflects the malware binary similarities can significantly lower the efforts in the identification and analysis of malware. In order to provide better malware classification heuristics, we perform clustering on our dataset and analyze the results in § V.

Code similarities across different malware families have been observed, implying the current naming conventions do not reflect actual similarities.

B. UNDERSTANDING MALWARE CHARACTERISTICS

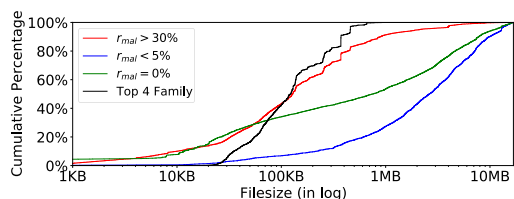
We further examine the characteristics of the binary samples. Specifically, we investigate their (i) file type (i.e., file extension), (ii) size, (iii) hash value, and (iv) packing method. Similar to the family distribution analysis, we base our analysis on the malicious ($r_{mal} > 30\%$) samples to describe the characteristics of malware. In addition, we include other samples in the malware file hash and size analysis. By also analyzing the characteristics of {potentially malicious, benign, unknown} samples, we draw a comparison among the samples with varying levels of r_{mal} . Such comparison allows us to highlight the prominent features of the malicious samples and capture the deliberate evasion techniques and distribution strategy of malware distributors today.

1) MALWARE FILE EXTENSION ANALYSIS

Table 3 shows the top 5 *true* file types of the collected malicious ($r_{mal} > 30\%$) samples and their advertised file types. The first row (x-dosexec, x-executable, msword...) shows the *true* file type of the distributed samples, and below are the filenames as shown in the URL or HTTP header. There are two places where the file format served in the URL is advertised: the first is the filename specified in the URL itself, and the second is in the content-disposition header in HTTP Protocol. Besides the samples shown in the table, approximately 18% of the URLs did not specify their filenames, and 10% did not specify the filename even in the header. For this reason, we incorporated *libmagic* python library in our collection system to automatically identify the true format. In terms of the true file type of samples, the most frequently appeared file type was Windows executables (x-dosexec) which accounted for 52%. Linux executables (x-executable) followed with 17.25%. Notably, Microsoft Word document

TABLE 3. Top 5 file types and their advertised file types of malicious binary samples ($r_{mal} > 30\%$). Malware are often in disguise as other file type than its true type.

x-dosexec (52.07%)	x-executable (17.25%)	msword (9.66%)	text/plain (8.37%)	text/html (5.91%)
.exe (60.0%)	No extension (50.9%)	.doc (76.8%)	.exe (96.6%)	.exe (85.3%)
.dll (4.6%)	.x86 (4.1%)	.b (2.2%)	.bat (1.8%)	.dll (6.4%)
Unknown (2.6%)	.mips (4.1%)	.c (1.1%)	.ps1 (1.2%)	.doc (5.2%)
.gz (2.6%)	.mpsl (3.8%)	.docm (1.0%)	.dll (0.2%)	.docx (2.2%)
.b (2.2%)	.arm7 (3.7%)	.a (1.0%)	.doc (0.1%)	.bat (0.8%)

**FIGURE 5.** Cumulative distribution of file size.

placed in third with 8.37%. We found that the file extensions in the collected URLs often do not match the true file format.

The purpose of hiding file types in distributing malware samples is usually to bypass rudimentary intrusion detection systems. A correct extension is not required for the correct execution since the droppers can arbitrarily rename the downloaded payload.

Automated filtering based on the advertised file extensions can be easily bypassed.

2) MALWARE DISTRIBUTION FOR IOT DEVICES

We identified that at least 11.6% of the malicious ($r_{mal} > 30\%$) samples labeled as Linux malware (i.e., x-executable) are compiled for non-x86 architectures in our dataset, as you can see in Table 2. A common practice we observed was that the attackers cross-compiled the same malware source code to multiple architectures, such as MIPS and ARM. Additionally, we observed that the attackers serve the same malware compiled for various architectures at a single distribution site in our dataset. We suspect that Linux-based IoT systems are becoming the target of interest for attackers. Notably, the Mirai malware had the most non-x86 architecture executables, which are known to target IoT devices [49].

IoT-targeting malware is still in active distribution as reported by previous works [50], [51].

3) MALWARE FILE HASH ANALYSIS

We found that many collected malware samples with different hash values, in fact, contain identical code contents. In the file hash analysis, we analyzed the samples with all ranges of r_{mal} . Notably, we find approximately 47% of malicious binary samples were duplicates with arbitrary data embedded at the end of the binary (i.e., *binary overlay*) inserted to yield a unique hash value. This suggests that the malware distributors are making conscious efforts to evade simple hash-based detection, which is a common detection technique found in the wild [52]. Unknown samples also have shown a high

TABLE 4. Number of binary samples with overlays and number of newly identified duplicates through strict hashing method.

Classification	w/ Overlay	Duplicate Count	Duplicate Percentage
Malicious	9,495	4,463	(47.0%)
Potentially malicious	1,489	85	(5.71%)
Benign	2,827	50	(1.77%)
Unknown	6,771	2,611	(38.56%)
Total samples	20,582	7,222	(35.09%)

duplicate percentage. Since we establish ground truth with VirusTotal's multi-vendor reports, we restrain from making a presumptuous conclusion in the context of this analysis. However, we revisit the unknown samples in § V and show that many unknown samples have high file content similarity to malicious samples in our clustering.

In order to better identify *unique* samples, we devised a hashing technique that we call *strict hashing*; we exclude the executable file headers and auxiliary sections. By only including the sections that are vital to executing the program (e.g., code and data sections), we were able to nullify adversary strategy to yield a unique hash value for each malware binary. We automate the process through a python script incorporated into our data collection system to trim the binaries before hashing.

Using strict hashing, we evaluated the uniqueness of the collected 99,312 binary samples. Table 4 shows the duplicate samples discovered through strict hashing. We found that the 20,582 samples had the aforementioned overlays. We also observed that the malicious file has a definite tendency toward duplication (i.e., just appending dummy data after the executable files) compared to the benign files, as 47% of the malware with overlay data was duplicated, and 1.77% of benign files did. One interesting result is that unknown files showed more tendency toward duplication with malicious files, suggesting that unknown files may contain sufficient amount of malware in the wild.

We further evaluated the collected binary samples to measure the prevalence of such practice in malware distribution. This suggests that mitigating detection with simple modifications is popular in the wild. In addition, these patterns are more common in malicious and potentially malicious files compared to benign files, meaning that our hashing scheme can detect the trivial files with low false positives. Moreover, we try to detect a known malware set based on strict hashes, whose hash is not known to VirusTotal. To this end, we check if strict hash of unknown samples are included in the known malware set and not included in benign malware set. As a result, there were 1,755 out of 6,771 (26%) unknown samples

had overlay data with malicious samples. Our analysis suggests that our way of hashing binary samples can eliminate the amount of hashing or binary analysis for a significant portion of unknown binaries.

Padding malware binaries with overlays to yield a unique hash values, is a prevalent practice in malware distribution today.

4) MALWARE FILE PACKER ANALYSIS

Malware is often *packed* to evade signature-based detection and raise the bar of the malware analysis. We measure the use of packing in the collected malicious ($r_{mal} > 30\%$) binary samples in our dataset.

Table 5 shows the distribution of the identified packing methods. We use F-PROT and PEiD packing detection provided by the VirusTotal attributes to detect and classify the packing methods. The top 5 prevalent malware packers were UPX, BobSoft, INNO, ASPACK, and NSIS, and frequent packers are the same for PE and ELF. Overall, about 27% of PE malware samples and 6.7% of ELF samples were detected with those packer detectors. Contrary to our expectation, well-known COTS (Commercial Off-The-Shelf) packing tools were not mainly used.

To further investigate the 73% of the samples that are supposedly not packed, we perform manual analysis on randomly selected 30 samples with distinct strict hash values. Unfortunately, checking whether each file is packed requires extremely high cost of manual reverse engineering, which forces us to resort to random sampling. The sample size is set according to the Central Limit Theorem, which proposes the minimal sample size of 30 for representing the whole distribution. As the analysis result, we found that none of the analyzed samples were packed. While we could not confirm our observation across all the samples, our best effort method allows us to make a statistical conjecture that a large portion of the samples are not packed. Also, we could observe the use of well-known packing tools, for instance, the top 5 observed in our dataset. This shows that the adversary has chosen to use simpler methods to generate malware with unique hashes and generate a large bulk of binaries that are essentially identical programs containing small differences. We surmise that this is due to the decreased value of malware implementations since the many popular malware families are open-source, as in the case of Mirai [49].

Contrary to previous beliefs, the practice of packing malware binaries has diminished. This is possibly because many popular malware families are open-source nowadays, and their code structure and behavior are well-known.

5) BINARY FILE SIZE ANALYSIS

In the Figure 5, we show a cumulative graph of file size with respect to the samples of all r_{mal} ranges ($r_{mal} > 30\%$, $r_{mal} < 5\%$, $r_{mal} = 0\%$), and the file size of top 4 malware families. We can find clear differences among the ranges. First, file sizes of malicious ($r_{mal} > 30\%$) samples are concentrated

TABLE 5. Packer usage statistics for malicious samples.

Packer(PE)	Percent	Packer(ELF)	Percent
UPX	1480(8.52%)	UPX	192(2.07%)
BobSoft	628(3.62%)	INNO	78(0.84%)
INNO	625(3.60%)	BobSoft	75(0.81%)
ASPack	409(2.36%)	NSIS	70(0.76%)
NSIS	389(2.24%)	ASPack	49(0.53%)

around 100KB, and 50% of malicious samples' file sizes were within 70KB ~ 600KB. Benign samples are uniformly distributed compared to other file types in terms of file size. And file sizes of the samples with ($r_{mal} < 5\%$) were relatively large compared to the other two classes. Interestingly, the file size of the top 4 families was even more concentrated than other classes, more than 50% of samples sized within 50KB ~ 200KB and 90% of samples within 30KB ~ 500KB. This result reveals that there are clear preferences in malware file sizes, which is mainly a few hundreds of kilobytes, and we believe that this is the strategic choice of recent attackers that optimizes the distribution of malware samples.

C. IN-DEPTH CASE STUDIES

1) EVASIVE .NET DOWNLOADER

Analyzing malware samples, we found a peculiar sample that was not detected by most AV vendors. This file is a simple downloader implemented in Windows PowerShell and had been actively distributed through *jplymell.com*. Interestingly, only one AV vendor(Symantec) detected the downloader. The file named `ps.ps1` downloads the next-stage payload, `applepeg.jpg`, from the same site and executes it. As an obfuscation method, the file randomly changes the script itself by inserting meaningless spaces and tabs and also encoding itself with base64.

The use of PowerShell with the particular obfuscation seemed to be one of the most efficient way to bypass AV systems.

2) MULTI-STAGED MALWARE USING CUSTOM PACKER

Emotet, the second-largest common malware on the internet, is delivered in a multi-stage manner and uses a custom packer. In this case study, we analyze a sample of the Emotet malware¹ active at the last day of our collection, starting from their initial compromise method, a Microsoft Word document embedding the malicious macro script.

First, opening the Word document shows a decoy image file, which tricks users into enabling the macro. When the user enables the macro, an obfuscated script assembles base64 encoded PowerShell scripts and spawns new processes. This obfuscation is hard to de-obfuscate or emulate since they split the script and store the chunks into various ways. For example, some script chunks are stored in the script itself, sometimes stored in an encoded form, or stored in GUI metadata, i.e., in the description attributes of a hidden

¹SHA256: 367beb7944831570410dcff59d7e8b2d5cf1074dd1ca52dee29f0dfc9785bfdd

text-box object. These various obfuscation techniques bypass the static analysis system, which emulates the script itself since the payload is stored out of the script and stored in Word document metadata.

After the macro spawns a new PowerShell process, the PowerShell script tries to download the next stage malware from four different C&C URLs, which is hard-coded as the PowerShell script. They check the size of the downloaded sample, so dynamic analysis sandbox with emulated network environment will fail to reveal the script's malicious behavior as they only check the validity of the network-delivered file. Interestingly, the malware distribution URL within the PowerShell script is different from where its origin macro document is distributed. This means that they have at least five different malware distribution sites to distribute an actual malware sample.

Next, the downloaded sample is custom-packed as a Windows PE file with various anti-analysis tricks. For example, they try to trick emulation-based detection by adding opaque predicates and heavy loops, as well as traditional VM detection techniques. Some of them use an anti-analysis trick called process-hollowing [53], which injects an unpacked code into other benign processes. As malicious behaviors spread through other benign processes that are not meant to be tracked, it makes hard to track the exact behavior of the malware. Furthermore, malware nowadays extensively uses cryptographic methods to secure its C&C channels, such as HTTPS and public-key cryptography that validates received information. Existing analysis technique based on network traffic emulation is hard to trick malware since it is hard to forge a message when the private key of the C&C infrastructure is unavailable.

Through our in-depth analysis of custom-packed malware samples, we identify the obstacles of diagnosing multi-staged malware, (i) various languages (i.e., word macro script, PowerShell script, binary codes), (ii) multiple malware distribution URLs in a single campaign, (iii) custom packers that evade automatic analysis, and (iv) robust communication channels using cryptography. These suggest that an integrated, robust analysis system must be researched in order to analyze and prevent recent multi-staged malware.

Another interesting feature of this malware is that the malware developer has actively employed diverse techniques to evade malware detection techniques. For example, *jplymell.com/extin/neutros.hhk* was alive from 14 January 2020 to 06 April 2020, distributing 21 unique malware samples. Interestingly, they updated and rebuilt their binary almost every day before 11 February 2020, and they did not update the binary sample² afterward. It seems that the attackers behind this campaign continuously updates their binaries and eventually settled to a successful one. This highlights the need for watching URL as well as binary to actively defend

²SHA256: 2270746c7bfdd89384f62c7734af07e04627a74dae292de987c7af9bdd8094fb

TABLE 6. Malware sample coping with distributing infrastructure takedown.

Date	URL	Tags
2020-02-02 23:52	/applepeg.jpg	
2020-01-07 07:26	/rootweb/applepeg.jpg	QuasarRAT
2020-01-06 18:06	/applepeg.jpg	njRAT
2019-11-20 11:40	/xmond/xop.exe	ImminentRAT
2019-08-20 13:10	/mail/smartapp.jpg	ImminentRAT
2019-08-20 13:09	/dmc/ps.ps1	
2019-05-20 18:24	/dmc/CLVIEW.exe	ImminentRAT
2019-02-13 17:41	/dmc/ImgFilePDF876...	

against such attacks, before they get improved enough to bypass automatic detection methods.

Also, manual analysis using VirusTotal passive DNS and Graph API reveals that an attacker behind this URL frequently changes the DNS record of the domain. The domain was registered in September 2019 and assigned to 23 distinct IP addresses. Some IP addresses were resolved to benign websites that likely use hosting services, or they might hijack other servers and change the domain to a hacked server. In addition, other malware samples distributed from *jplymell.com* are also connected to *linkadrum.nl* and *pent-housefb.org*, so there is a high probability that the attacker also controls those servers. This domain had distributed malware for more than a year, which means that AV vendors had enough time to defend against those malware samples, but the attacker had effectively evaded antivirus detection.

In addition, we checked the URLHaus reports to confirm the changing trend in the malware family, which is shown in Table 6. The domain *jplymell.com* used ImminentRAT until November 2019, and it moved to QuasarRat. After the takedown of the ImminentRAT infrastructure by Europol, it quickly found an alternative and continued the campaign [54].

Through this in-depth analysis, we identified multiple characteristics of the advanced attack campaign, (i) Assigning multiple IP addresses with a domain, (ii) Using multiple domains in a campaign, (iii) Updating its binary files frequently while the distribution URL remains unchanged, (iv) Swapping malware family from time to time. However, all of these attacks can be grouped by their primary asset, malware distribution URL, suggesting the need for considering malware downloading URL as well as malware sample itself.

V. CLUSTERING FOR BETTER MALWARE CLASSIFICATION HEURISTICS

Malware binary clustering provides clear advantages in the fight against malware distributors. As we discussed in § IV, being prolific is one of the main strategies of malware distributors. Clustering malware binaries allows researchers and industry to obtain a comprehensive view on the currently distributed malware. Also, up-to-date malware clusters allow prioritizing of limited man-hours into new variants of malware that require attention. Previous works [8], [55] also pointed out that malware clustering can cover this issue.

We perform clustering on our malware binary dataset to create classification heuristics that are based on the *file similarities*. Our clustering seeks to illustrate the possible relations among the samples in our dataset at a different angle, aside from the AV vendor labels, since the malware labels from AV vendors report conflicting classification and allow many binary samples to go undetected. Towards this goal, our clustering includes all samples in our dataset (potentially malicious, benign, and unknown samples).

The clustering yielded 868 clusters with more than five samples, involving a total of 44,104 samples. The largest cluster had 1911 samples. In the rest of this section, we explain our clustering method and discuss the result and its implications.

A. CLUSTERING BASED ON FILE SIMILARITIES

Recently, AV vendors have suffered from flooding malware samples into their analysis pipeline, making it hard to analyze all the collected data even with the automated dynamic analysis. Thus, they adopt various techniques such as static analysis or machine learning based detection or clustering to reduce the size of the corpus. Clustering enables us to track the variants and provides a simple yet effective method for identifying unknown malware samples. Based on clustering results for unknown files, we can also designate and manage the “malware candidate” group that needs to be monitored among them.

1) CLUSTERING METHOD

We evaluate the clustering approach's effectiveness using our malware samples. We use TLSH [56], which produces a distance score when given two binaries using *locality-sensitive hashing schemes*, as in a few previous works [56], [57]. The distance score has a range of $0 \sim 300$, where 0 indicates identical files, and the higher score indicates larger differences. We also adapted *agglomerative hierarchical clustering*, which first assigns each sample to a singleton cluster, then recursively merges two adjacent clusters. More specifically, we used a *single-linkage metric*; two closely located clusters when the distance between the two is less than the given threshold. We applied a threshold of 80 based on the empirical results from previous work [57].

2) CLUSTERING RESULT

We found 868 clusters with more than five samples and identified the biggest clusters with their component families. Table 7 shows our clustering result. The top 10 clusters were mainly Emotet, Mirai, and Trickbot samples. There were 44,104 samples in those 868 clusters, and we could skip more than 40% of the samples and focus on other samples that are not similar to known malware and thus require careful analysis. This suggests that the clustering approach works on the recent malware sets, providing a method to prioritize the analysis pipelines. Also, we can find that sufficient amount of unknown samples are included in each cluster. In total, 23% of unknown samples are clustered.

Unknown samples that are clustered with malware are suspected to be malware that is undetected or in pre-distribution stage.

We randomly selected five samples from potentially malicious samples in each cluster and manually analyzed them as noted in the remark in Table 7. Due to the high manual effort required for reverse engineering of each sample, we could not verify all samples. However, all five samples were found to be a variant of the dominant AV-vendor-confirmed malicious families in the cluster. While we can not assume that all potentially malicious samples in the dominantly malicious clusters are malicious, we argue that our clustering allows the researchers and task force in the field to narrow down on the samples that are *unknown*, but worth manual reverse engineering efforts. While there were clusters with uncertainty, such as the 6th cluster having 13.5% of Fareit and 4% of Garamue, which requires further investigations, the clustering approach worked well for all benign, potentially malicious, and malicious clusters.

Also, we figure out which families are well clustered using our approaches. For example, Trickbot was dominant in cluster #4, accounting for 99% of known samples. This means that Trickbot family is cluster-friendly (i.e., well clustered using similarity hash). Cluster-friendly families can be said to be families with many overlays and an active generation of variants. Therefore, if an unknown file is incorporated into a cluster mainly composed of files of a clustering-friendly family, we can guess with high confidence that it is malware that belongs to the family. To find out all cluster-friendly families from our dataset, we searched for families that were accounted for more than 80% of known samples in any cluster. As a result, 42 families were identified to be cluster-friendly, and Table 8 shows the full list of them. Given that 20,486 samples are classified into these 42 families, which is 38.2% of reported samples and 54.9% out of malicious samples, we confirmed that clustering works on recent and large numbers of distinct families.

3) UNDERSTANDING MUTATION STRATEGY

The clustering result shows us that an abundant number of variants exist in the wild, which are similar in terms of their bytes level characteristics, and these are easily detectable using similarity hash. Therefore, the next research question is how the similar samples inside a cluster are actually modified. In order to answer this question, *binary differences* between the samples found in each cluster are needed. Thus, we iteratively applied the longest common sub-string algorithm to the pairs of samples. Also, we calculated the entropy of files' bytes contents to find the relationship between the density of information and the frequency of modification among the file contents. If frequently modified parts have a small density of information, then it can be inferred that the attackers reuse existing malware, only changing useless parts. We only select binary difference results with more than 50% matches and mark which parts in the malware are frequently shared and

TABLE 7. Top 10 largest clusters and their component families.

Size	Family	Result	Examples of Potentially Malicious Samples in Cluster
1911	emotet	69.18% mrwx 3.72% potentially malicious 0.37% unknown 19.88%	Word macro malware
1851	mirai	51.22% gafgyt 23.72% potentially malicious 1.30% unknown 22.91%	Mirai (ARM)
1744	trickbot	77.64% wacatac 0.06% potentially malicious 0.06% unknown 22.25%	One Trickbot sample with lower r_{mal} .
1298		potentially malicious 0.15% unknown 99.85%	Two normal word documents.
1168	ursnif	14.64% ursnifdropper 0.94% potentially malicious 0.00% unknown 83.90%	-
1133	fareit	13.50% gamarue 3.97% potentially malicious 3.97% unknown 20.74%	2 out of 5 was malicious.
1012	trickbot	75.40% ursu 0.20% potentially malicious 0.00% unknown 24.31%	-
987	trickbot	0.41% obfuse 0.10% potentially malicious 3.75% unknown 95.74%	Word macro spreading Trickbot
811	class	3.21% minimal 2.84% potentially malicious 0.00% unknown 18.74%	-
761		potentially malicious 52.69% unknown 47.31%	Normal Linux executables

modified between two malware samples. We then randomly selected 50 pairs in each cluster in total and found that the average overlapping part was 83.2%. This means that similar malware, regardless of its family, shares more than 80% of its contents, modifying a relatively small part of the sample. Figure 6 shows the most frequent part that the malware author modifies when generating variants. This clearly shows that the attackers mainly modify two parts. First is a small part at the front part, where header contents are located. The other part is the end part, where resources or overlay data are located. Also, the entropy of the file content, which is denoted by the red line in Figure 6, suggests that the part with higher entropy, i.e., having more information than the other part, remains unchanged, and lower entropy parts are more likely to be modified when generating malware variants.

Malware authors generate variants by simply modifying the front and end parts of the files.

4) TRIVIAL CHANGES ARE THE NORM IN MALWARE DISTRIBUTION

We found a malware distribution domain³ that distributed 2,756 unique malware samples and AVCLASS result suggests that these samples were Trickbot family. By calculating binary differences on these samples, we figured out that most of the files are trivial variants: generated by appending dummy bytes at the end of the original file. In more detail, there were actually two strains of malware, the first one was around 300KB, and the other one was around 450KB. We found that all samples, each for 1,744 and 1,012, were actually generated by applying trivial modification to the two original samples. Appended dummy data were 0 to 1,114 bytes and 0 to 601 bytes at the end of the file, and the entropy of added part was 3.1 on average. These samples were 4th, 8th ranked clusters from the previous subsection and found that clustering performed well to these trivial changes in the malware variant generation.

Trivial variants are frequently found in the wild: malware authors simply put dummy data with lower entropy at the end of the malware, to generate an unseen malware sample in terms of the same hash.

TABLE 8. Clustering-friendly families: at least one cluster contains more than 80% of the family in above.

agobot, alien, autoit, cliptoshuffler, constructor, csdimonetize, dofoil, downloadguide, duote, duqu, emotet, explosive, gafgyt, gandcrab, gozi, hajime, hebogo, himera, installcore, iroffer, khalesi, mikey, mirai, nanocore, neoreklami, nethief, qjwmonkey, razy, relevantknowledge, sinowal, spigot, swrort, theef, tinyloader, trickbot, tsunami, ulise, ursnif, utilpack, vebzenpak, wget, xorddos, zenpak

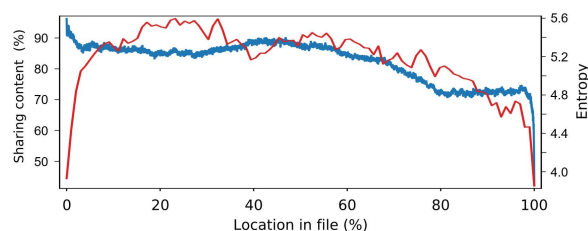


FIGURE 6. binary diff result and entropy of similar malware samples.

5) TOWARDS MALWARE DETECTION WITH CLUSTERING

We point out that AV vendors and file analysis services cannot sufficiently reflect the behavior of the binary into the classification of malware, as they sort a file into known & unknown only with their policy hashes. Furthermore, the naming scheme of the malware varies around AV vendors, as VirusTotal reports show. We argue that the varying naming scheme around AV vendors and hash-based malware detection make it hard to track fast-mutating malware variants and detect files that are in the pre-distribution stage. We suggest that similarity-based malware clustering should be applied around AV vendors to solve these problems. To do so, sharing malware samples through vendors and the public would be needed to enable comparing the similarity of malware around the ecosystem getting compared.

Clustering can provide an unified standard on classifying malware, and also help to detect malware files in pre-distribution.

B. DISCUSSION

Our clustering seeks to provide an alternative to the traditional malware labeling practice.

1) CLUSTERS AS MALWARE CLASSIFICATION LABEL

We point out that the criteria for classifying malware should focus on reflecting their behavior patterns and binary similarities. The current labeling scheme differs upon AV vendors

³<http://www.mitsui-jyuku.co.jp>

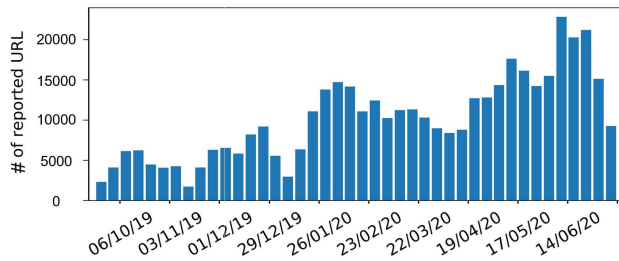


FIGURE 7. The number of reported URLs in our crawling (per week).

with truly high variance, and we argue that this makes it difficult to keep track of malware mutation, which eventually leads to the reduction of the attacker's burden. Therefore, we assert that binary-similarity based cluster should be the standard for malware classification labeling.

2) BENIGN SAMPLES INCLUDED IN CLUSTERS

We found that malware samples that were previously classified as *benign* by the AV vendor reports were likely to be undetected malicious files. Even though the absolute number of benign samples in each cluster is a bit small, our static analysis of those files revealed that they were usually malware that had not been detected. Table 7 shows the type of malware that was included in the benign set. This implies our clustering approach can capture the malware that current AV vendors cannot detect.

3) TOWARDS ONLINE MALWARE CLUSTER

We plan to maintain our data collection system and clustering system such that the clusters are constantly updated. In such systems, users can query their own file to check if it is benign, or figure out the malware clusters near the file if it's not. Therefore, the users can get sufficient information on the files that are yet reported, also provide data to our clustering system.

VI. DISTRIBUTION NETWORK AND SITE ANALYSIS

We conduct a quantitative analysis to understand the current state of malware distribution. To this end, we dissect the statistics of the collected distribution site URLs in our dataset. Our analysis includes unique IP addresses, domain, and AS registration distributions, as well as URL lexical distributions of malware distribution sites. Through the analysis, we intend to present an overview of network infrastructures that the malware distributors exploit.

A. MALWARE DISTRIBUTION TREND

1) INCREASE IN DISTRIBUTION SITES

We observe a growth in the reported number of URLs during our data collection period (2019.06 ~ 2020.06). Figure 7 shows the growing number of reported URLs followed by their reports. The number peaked in May 2020, with more than 20,000 malware URLs, which is about four times more than that of June 2019.

2) DISTRIBUTION UPTIME

Furthermore, we found that the uptimes of URLs differ significantly between the malicious and benign samples. The average uptime of all reported URLs was 43 days. URLs that were confirmed to be malicious (i.e., $r_{mal} > 30\%$) had an average uptime of 15.45 days. The number is contrary to the average uptime of unknown (i.e., $\nexists r_{mal}$), which was measured to be 45.45 days. This clearly shows that malware distributors are forced to update their URLs to evade the real-time malicious URL blocking often deployed in end-point AV products.

Average uptime of malware distribution sites was 15.45 days.

B. DISTRIBUTION OVER NETWORKS

Additionally, we analyze the network aspects of the collected malware distribution site dataset. Since multiple distribution site URLs resolve to the same IP address, we first eliminate duplicates such that our dataset contains a unique IP address set. After the duplicate elimination process, the IP address set contains 13,798 unique IP addresses.

1) IP NETWORK DISTRIBUTION

Figure 8 presents the distribution of malware distribution sites over the IP address. Compared to the previous measurement work on specific families of malware [49], distribution sites in our dataset show a uniform distribution over the IP address space.

Our dataset of malware distribution sites in those ranges mentioned before is slightly more dense than other IP ranges, but they only cover 43.3% of the sites. This result suggests that malware distribution sites are evenly distributed, implying that URL blocking strategy based on specific IP ranges or ISPs would be infeasible.

When the malware distribution of the overall network is divided into B-class ranges, many B-class ranges include a small number of malware distributions sites (< 1%), as shown in Table 9. This augments our observation that malware distribution sites are uniformly distributed without a noticeable concentration on a specific range.

Malware distribution sites are uniformly distributed across multiple IP ranges, rendering defensive strategies that focus on specific IP ranges infeasible.

2) ASN DISTRIBUTION

We analyze the malware distribution in terms of *Autonomous System Number (ASN)*, the control unit of the network (Table 10). Even from the ASN perspective, as we expected, ASN in China is at the top of the rank for entire malware distribution. Among the Top-10 ASN of Malware distribution, AS4837, AS4134, and AS1325 are ranked at 1st, 2nd, and 5th respectively, with 22.5% of the entire malware distribution. The 1st in the rank, AS4837, was the most used with 45.8% of Gafgyt malware and was also used in 6.8% of Mirai. The 2nd, AS4134, was also used primarily as 12.8% of Gafgyt, and also AS132525 was used. As such, the vulnerability of

TABLE 9. Top-10 of B-class include malware distribution sites.

Total Malware (7820)	Mirai (755)	Emotet (2628)	Trickbot (19)	Gafgyt (2394)
111.42 (CN, 1.1%)	37.49 (EE, 9.3%)	166.62 (US, 1.6%)	150.95 (JP, 5.3%)	111.42 (CN, 3.0%)
37.49 (EE, 1.0%)	45.95 (HR, 8.3%)	145.14 (NL, 1.6%)	89.40 (FR, 5.3%)	162.212 (CA, 2.9%)
45.95 (HR, 0.9%)	116.114 (CN, 3.0%)	162.241 (US, 1.6%)	128.226 (US, 5.3%)	72.2 (US, 2.6%)
162.212 (CA, 0.9%)	185.172 (NL, 2.5%)	107.180 (US, 0.9%)	199.188 (US, 5.3%)	115.49 (CN, 2.3%)
111.43 (CN, 0.9%)	111.43 (CN, 2.3%)	205.144 (US, 0.8%)	198.54 (US, 5.3%)	123.11 (CN, 2.2%)
116.114 (CN, 0.8%)	111.42 (CN, 2.1%)	192.124 (US, 0.8%)	198.72 (CA, 5.3%)	123.10 (CN, 2.1%)
72.2 (US, 0.8%)	194.15 (DE, 2.0%)	217.160 (DE, 0.7%)	162.241 (US, 5.3%)	216.221 (US, 2.1%)
115.49 (CN, 0.7%)	45.14 (RO, 2.0%)	160.153 (NL, 0.7%)	149.255 (GB, 5.3%)	111.43 (CN, 2.1%)
162.241 (US, 0.7%)	176.123 (MD, 1.7%)	148.66 (SG, 0.7%)	174.142 (CA, 5.3%)	173.242 (US, 1.9%)
123.11 (CN, 0.7%)	45.148 (NL, 1.7%)	50.63 (US, 0.6%)	66.70 (CA, 5.3%)	182.127 (CN, 1.7%)

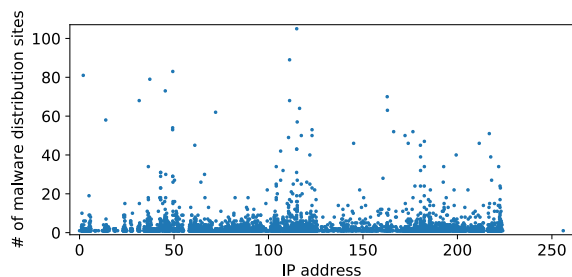


FIGURE 8. Distribution of distribution sites over IP address.

ASN in China is seen to still exist as in previous studies. And in the case of Gafgyt, it seems to be actively utilizing such vulnerability. Although Mirai includes some ASN in China of the Top-10 ASN, its proportion was little. Moreover, Emotet and Trickbot tend to exploit the United States and Canada or various countries, diverging from the ASN of China.

One-third of all malware is distributed by the top-10 ASN of the distribution site.

C. DISTRIBUTION OVER DOMAIN NAMES

The features extracted from domain names of the site of malware distribution can embed more valuable information than those from just simple IP addresses. The domains of our data set may not be directly generated by the malware distribution attacker or compromised sites. However, these domain names are involved in the malware distribution environment, and we expect this would help us to understand the characteristics of malware distribution behavior more clearly.

1) TLD LEVEL DISTRIBUTION

The first feature we can find from the domain is the TLD-level distribution of domain names. Table 11 shows the top-10 generic top-level domains (gTLDs) of top-4 malware, Mirai, Emotet, Trickbot, and Gafgyt. Except for Gafgyt malware, '.com' accounts for more than 40% of all Top-3 Malware. The proportion of '.ads1' ranks the 2nd, accounting for about 11.2%. Notably, the distribution sites with '.ads1' take 60.6% of all Gafgyt malware family distribution. This possibly implies that the Gafgyt distribution campaign was permred by a single group that simply reused their hosting services of choice. In addition, '.eu' domain in the Mirai's distribution is ranked 3rd with 5.8% interconnection of the

Malware distribution centered around Estonia and Croatia. Also, '.in' domain, which is the national domain of India, possesses 3.4% of Gafgyt's TLD distribution. Our observation confirms a previous report on the outbreak of Emotet malware in India in 2019 [58]. We also observe 56 malware URLs from 26 domains with government (.go, .gov) TLD that distribute malware. These sites are from China, India, Indonesia, Kazakhstan, Pakistan, the Philippines, Thailand, and Vietnam, and we suspect that those sites are compromised and used to distribute Windows malware.

Most malicious URLs had common TLDs (e.g., .com), but the use of the certain TLDs (e.g., .go, .gov) was only observed in certain regions.

2) URLs IN ALEXA

Also, we compare the Effective Second-Level Domains (e2LDs) against Alexa top sites. e2LDs allow us to understand the domain ownership than Fully Qualified Domain Names (FQDNs). As a result, there are 5,780 e2LDs in our dataset, and 13 domains are in the top 1000 sites, and 427 sites are in the top 1 million sites. Table 12 shows 13 sites from Alexa top 1000, and it includes qq.com, googleusercontent.com, discordapp.com. This means that malware distributors also abuse file-sharing features in common internet services (e.g., QQ, and Discord) as distribution sites. Hence, our observation calls for the integration of malware binary pre-screening features into such services.

Malware distributors also abuse file sharing features in common services as distribution sites. Binary screening can be deployed to such services for mitigation measures.

3) MALWARE FAMILIES AND DOMAIN RELATIONS

We also observe consciously diversified malware distribution in a single domain. We calculated the variance of r_{mal} within the family under the domain. subsection VI-D shows the number of files, the standard deviation of r_{mal} , and the malware families found in each domain sorted by the standard deviation of r_{mal} . The result shows that the highest r_{mal} variances are 0.197, and the most frequently discovered families are the gamarue, wacatac, lokibot, and agensla families. This result shows that malware distribution domains, each presumably used by a single distributor, distribute multiple types of malware families.

TABLE 10. Top-10 of ASN of malware distribution sites.

Total Malware (7820)	Mirai (755)	Emotet (2628)	Trickbot (19)	Gafgyt (2394)
AS4837 (CN, 15.4%)	AS213371 (NL, 9.3%)	AS26496 (SG, 6.9%)	AS22612 (US, 10.5%)	AS4837 (CN, 45.8%)
AS4134 (CN, 4.4%)	AS42864 (HR, 8.1%)	AS16276 (CA, 4.6%)	AS32613 (CA, 10.5%)	AS4134 (CN, 12.8%)
AS16276 (FR, 3.0%)	AS16276 (AU, 7.7%)	AS14061 (NL, 3.9%)	AS46606 (US, 10.5%)	AS26727 (US, 9.6%)
AS26727 (US, 2.9%)	AS54290 (US, 7.2%)	AS46606 (US, 3.0%)	AS7506 (JP, 5.3%)	AS132525 (CN, 6.8%)
AS132525 (CN, 2.7%)	AS4837 (CN, 6.8%)	AS16509 (US, 2.9%)	AS199653 (FR, 5.3%)	AS19465 (CA, 5.1%)
AS26496 (US, 2.6%)	AS132525 (CN, 6.0%)	AS13335 (US, 2.5%)	AS4190 (US, 5.3%)	AS16276 (US, 1.5%)
AS14061 (US, 2.0%)	AS24961 (US, 5.3%)	AS15169 (HK, 2.4%)	AS34931 (GB, 5.3%)	AS36352 (US, 1.3%)
AS46606 (US, 1.7%)	AS36352 (US, 5.1%)	AS24940 (DE, 2.3%)	AS16276 (CA, 5.3%)	AS42864 (HR, 1.1%)
AS16509 (SG, 1.6%)	AS206898 (NL, 2.5%)	AS45090 (VN, 1.9%)	AS29802 (US, 5.3%)	AS24961 (DE, 1.0%)
AS19465 (CA, 1.6%)	AS200019 (MD, 2.1%)	AS37963 (CN, 1.8%)	AS31034 (IT, 5.3%)	AS213371 (NL, 1.0%)

TABLE 11. Top-10 of TLD of malware distribution sites.

total	mirai	emotet	trickbot	gafgyt
com (39.2%)	com (47.6%)	com (43.8%)	com (52.4%)	adsl (60.6%)
adsl (11.2%)	net (11.8%)	in (3.4%)	jp (9.5%)	com (23.9%)
net (4.4%)	eu (5.8%)	org (3.0%)	co.uk (9.5%)	net (2.3%)
ru (2.8%)	org (3.5%)	net (2.9%)	live (4.8%)	dhcp (1.2%)
org (2.1%)	it (2.3%)	vn (2.7%)	edu (4.8%)	ru (1.2%)
in (1.9%)	xyz (2.1%)	ir (2.5%)	de (4.8%)	eu (1.0%)
com.br (1.6%)	de (1.8%)	ru (2.1%)	eu (4.8%)	pix (1.0%)
ir (1.3%)	io (1.4%)	com.br (1.9%)	icu (4.8%)	it (0.8%)
cn (1.3%)	ro (1.2%)	cn (1.9%)	org.uk (4.8%)	com.cn (0.8%)
vn (1.2%)	us (1.2%)	info (1.4%)	Unknown	rs (0.8%)

TABLE 12. e2LDs found in Alexa top 1000 sites.

Rank	e2LD	Counts
9	qq.com	1
55	wordpress.com	1
90	googleusercontent.com	128
190	sogou.com	1
218	discordapp.com	56
261	reimageplus.com	1
404	youdao.com	1
480	free.fr	4
504	t-online.de	1
574	tistory.com	1
577	mit.edu	2
736	sakura.ne.jp	7
748	secureserver.net	2

Malware distributing domains often distribute multiple types of malware families, and there is no distinct distribution site to malware family relations.

D. DISTRIBUTION OVER GEOGRAPHIC LOCATION

1) GEOGRAPHICAL LOCATIONS OF DISTRIBUTION SITES

The top-7 country result of the distribution site in our dataset shows similar results as a previous work [4], except that the rankings have slightly changed. Table 14 shows detailed statistics from our dataset. Notably, the Gafgyt malware family is overwhelmingly prevalent in China. On the other hand, the United States have a large portion in the total number of the top-3 malware (Mirai, Emotet, Trickbot) malware family distribution. Compared to the concentration of ASN, the results differ slightly.

E. DISTRIBUTION SITE ANALYSIS DISCUSSION

Our analysis provides a foundation for further studies on understanding the malware distributor strategy in choosing and using the acquired distribution sites. We observed that the distribution sites show a uniform distribution in IP addresses

and domains. We argue that the distributors make such a conscious diversification effort, given the AV vendors' continuous reactive defensive measures. Also, the diversified malware family outflux from a single domain is certainly not cost-efficient. Purchasing malware generation toolkits for different families and mutating them with the strategies we found in § IV, and replacing binaries with a short time period (15.45 days) are daunting tasks. We conclude that distribution sites cannot be a reliable indicator for the maliciousness of served files or distributed malware family type. This is why we did not include distribution sites in our clustering, shown in § V. We believe that hosting sites must be involved with the defensive effort in malware distribution prevention. That is, file screening policies based on the real-time and up-to-date malware databases in hosting services would significantly raise the cost of distributing malware.

VII. INSIGHTS AND IMPLICATIONS

In this section, we discuss the implications from our findings and suggest future directions for the research community and industry involved in malware defense.

A. OPEN-SOURCE AND CROSS-PLATFORM MALWARE

In our analysis, we observed a large corpus (11.6%) of malware that are built for non-x86 architectures (e.g., ARM and MIPS), as shown in Table 3. These malware samples are predominantly Mirai variants. We surmise that the malware distributors are conveniently cross-compiling the now open-source Mirai to multiple platforms. Our findings confirm the substantial malware threat in IoT devices reported in previous works [49]–[51]. We expect IoT malware distribution to continue to grow, and the current trend – the multi-target compilation of the same malware – will continue. This phenomenon calls for the development of cross-platform

TABLE 13. Top 15 netloc with various malware "SINGLETON-n" means n singleton family were found on the site.

URL	n_{file}	σ	AVClass Family under sites
apbfiber.com	337	0.197	regotet ursnif
gazpromstaff.com	11	0.184	agensla autoit gamarue lokibot redcap scar scarsi wacatac SINGLETON-1
memberteam.works	537	0.179	gozi ursnif ursnifdropper wacatac yakes SINGLETON-68
mitsui-jyuku.mixh.jp	2205	0.174	trickbot ursu wacatac SINGLETON-1
admaris.ir	25	0.167	chisburg darkkomet fareit gamarue lokibot scarsi wacatac zusy SINGLETON-2
c32.19aq.com	30	0.166	agen backex casur exploiter fpipe gendal gsecdump mimikatz moderate opencandy parite pwdump saminside sqltool toksteal yakes SINGLETON-3
spartvishltd.com	27	0.165	agensla buerak fareit gamarue high hploki injects lokibot predator wacatac
abass.ir	43	0.165	agensla agenttesla fareit formbook gamarue high lokibot nanobot nanocore noon remcos spybotnet SINGLETON-1
cdn.discordapp.com	56	0.162	addrop atraps autoit avemaria badjoke bladabindi delf discord esulat gamarue high installcore joke makoob mediaget moderate msilperseus nanocore netwiredrc noon occamy onlinegames pyfatget quasar razy rrat schoolgirl subti vebzenpak wacatac wannacry SINGLETON-7
logroom.top	41	0.161	agensla autoit bladabindi fareit formbook gamarue high lokibot remcos vebzenpak wacatac
powerlogs.top	13	0.159	e99c3c72653d fareit gamarue lokibot remcos wacatac
jpymell.com	19	0.157	azorult downeks high msilperseus nanobot noon quasarrat SINGLETON-2
robotrade.com.vn	23	0.156	agensla bladabindi cryptos high nanobot noancooe racealer
pivotpower24.com	42	0.153	autoit
jload08.xyz	11	0.152	clipbanker coins cryptbot high occamy ursu

TABLE 14. Top-10 of Country of malware distribution sites.

Total	Mirai	Emotet	Trickbot	Gafgyt
China (24.6%)	United States (26.4%)	United States (29.9%)	United States (45.0%)	China (66.9%)
United States (23.3%)	China (14.2%)	Germany (7.5%)	Canada (15.0%)	United States (14.6%)
Germany (4.3%)	Netherlands (13.3%)	China (6.0%)	Japan (10.0%)	Canada (5.9%)
Russia (3.8%)	Estonia (7.7%)	Vietnam (5.8%)	Germany (5.0%)	Netherlands (2.0%)
Netherlands (3.6%)	Germany (7.4%)	France (4.8%)	United Arab Emirates (5.0%)	Russia (1.5%)
Canada (2.9%)	Croatia (7.3%)	Singapore (4.6%)	France (5.0%)	Germany (1.3%)
France (2.8%)	France (3.5%)	Netherlands (4.2%)	Italy (5.0%)	Croatia (0.9%)
Vietnam (2.6%)	Canada (3.3%)	India (4.0%)	Ireland (5.0%)	France (0.9%)
India (2.3%)	Romania (2.2%)	Russia (2.6%)	United Kingdom (5.0%)	Estonia (0.7%)
Singapore (2.3%)	Republic of Moldova (2.1%)	Hong Kong (2.1%)	Cambodia (0.7%)	

malware similarity evaluation techniques such as the work by Xu et al. [59], [60]. Such techniques will render the identification of new IoT malware variants significantly more efficient by allowing identification of their traditional computing device (e.g., desktops) counterparts that are likely to be already in the malware databases.

B. CALL FOR STANDARDIZED MALWARE FAMILY CLASSIFICATION CONVENTION

We found frequent malware binary label discrepancies across multiple AV vendors. This poses difficulties in establishing the ground-truth in collecting and analyzing malware binaries. It leaves researchers no option but to rely on the tools that aggregate different family naming convention of AV vendors, such as AVCLASS. This problem has been raised by various previous works [61]–[63] on malware analysis, but still remains unchanged. Moreover, we found that current malware naming scheme does not reflect the similarities among the families. We found that two malware families with distinct names had over 60% file similarities (Figure 4). Also, our clustering based on TLSH (§ V) shows actual similarities in binary contents generated clusters containing binaries with different family classification but with high file similarities.

This shows that the current malware family labeling system does not reflect differences in the binary properly, making it hard to track variants of fast-changing malware families. We expect that prescribing a standard for classifying malware families would reduce the cost of analysis for AV vendors.

There have been some efforts on enacting a standard malware naming convention, such as CARO [64] and MAEC [65], but they are yet to be widely adopted. Alternatively, researchers attempted to cluster malware with their content or behaviors [8], [45], [55] or employ supervised machine learning [66]–[68]. Towards a standardized malware family classification convention, we expect that an objective comparison of the existing classification methods will significantly aid the researchers in the field. Also, the AV vendors should cooperate in reporting consistent and standardized labels for malware.

C. COUNTERING ADVERSARY'S EVASIVE STRATEGY

As we report in this work, malware generation and distribution is a mature industry. Fighting malware has proved to be a daunting challenge through decades of the arms race. Hence, practical defense measures must raise the cost of malware distribution and damage the profit. Our binary

analysis indicates that the malware distributors can regenerate malware binaries with unique hash values at a rapid rate. As we discussed, malware binaries are deployed with overlays that add just enough differences to yield a unique hash value. This indicates that the adversaries are well aware of, and can economically evade, the traditional hash blacklisting approach.

On the other hand, the number of malware distribution URLs is relatively limited (13,798 unique IP addresses). Also, the average number of uptime data for confirmed-malicious ($r_{mal} > 30\%$) distribution sites was 15.45 days (from the time it was first identified as malicious). This is contrary to the average uptime of unknown (i.e., $\neq r_{mal}$), which was measured to be 45.45 days. The adversary makes conscious efforts to diversify the distribution sites, as we report in this work. Nevertheless, our findings indicate that URL blacklisting would generally apply significantly more pressure on the adversary than binary hash blacklisting.

As such, we expect that research efforts that propose more efficient proactive distribution site detection [69]–[71] will be valuable to the researchers and practitioners in the field.

VIII. CONCLUSION AND FUTURE WORK

In this study, we presented a large-scale and comprehensive analysis of the current status of malware distribution. We built a malware data collection system for the analysis and collected 99,312 malware binary samples and 38,659 malware distribution sites over 287 days. We dissected the statistics of the collected malware and their distribution sites to identify the current trend of malware distribution.

Through multifaceted analysis on our malware binary samples as well as peculiar outliers, we identified the general trends in the current malware binaries and the current evasion strategies employed by the adversary. Motivated by the current malware family classification convention that is often not in consensus among different reports, we presented our malware binary clustering based on the actual differences in the binaries and its result. We spotted the trivial modifications through the clustering approach while generating malware variants. We also explored the possibility of detecting malware in the pre-distribution stage through clustering by manual analysis on unknown samples in the cluster. Regarding the distribution sites of the malware binaries, we revealed that adversaries are consciously trying to diversify the distributed malware in a single site.

Based on our analysis, we provided insights and implications from our findings and proposed possible countermeasures on malware distributors' evasive strategies. Among the suggested directions that may contribute to the ongoing efforts against malware in this work, we plan to further advance the malware family classification method and convention based on the proposed clustering method. In more detail, we plan to explore the feasibility of a standardized similarity score convention for potentially malicious binaries based on actual file similarity that can supplement the limitations of the current malware classification.

In all, we provided a comprehensive overview of the current state of malware and its distribution, and a summary of insights for researchers and taskforce in the field.

REFERENCES

- [1] J. Caballero, C. Grier, C. Kreibich, and V. Paxson, "Measuring pay-per-install: The commoditization of malware distribution," in *Proc. USENIX Secur. Symp.*, vol. 13, 2011, pp. 1–16.
- [2] K. Thomas, J. A. E. Crespo, R. Rasti, J.-M. Picod, C. Phillips, M.-A. Decoste, C. Sharp, F. Tirelo, A. Tofigh, M.-A. Courteau, L. Ballard, R. Shield, N. Jagpal, M. A. Rajab, P. Mavrommatis, N. Provos, E. Burszstein, and D. McCoy, "Investigating commercial pay-per-install and the distribution of unwanted software," in *Proc. 25th USENIX Secur. Symp. (USENIX Secur.)* Austin, TX, USA: USENIX Association, Aug. 2016, pp. 721–739. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/thomas>
- [3] P. Kotzias, L. Bilge, and J. Caballero, "Measuring PUP prevalence and PUP distribution through pay-per-install services," in *Proc. 25th USENIX Secur. Symp. (USENIX Secur.)* Austin, TX, USA: USENIX Association, Aug. 2016, pp. 739–756. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/kotzias>
- [4] C. C. Iff, Y. Shen, S. J. Murdoch, and G. Stringhini, "Waves of malice: A longitudinal measurement of the malicious file delivery ecosystem on the web," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, Jul. 2019, pp. 168–180.
- [5] B. J. Kwon, J. Mondal, J. Jang, L. Bilge, and T. Dumitras, "The dropper effect: Insights into malware distribution with downloader graph analytics," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2015, pp. 1118–1129.
- [6] X. Ugarte-Pedrero, M. Graziano, and D. Balzarotti, "A close look at a daily dataset of malware samples," *ACM Trans. Privacy Secur.*, vol. 22, no. 1, pp. 1–30, Jan. 2019.
- [7] O. Alrawi, C. Lever, K. Valakuzhy, K. Snow, F. Monrose, and M. Antonakakis, "The circle of life: A large-scale study of the IoT malware lifecycle," in *Proc. 30th USENIX Secur. Symp. (USENIX Secur.)*, 2021, pp. 3505–3522.
- [8] X. Hu, K. G. Shin, S. Bhatkar, and K. Griffin, "MutantX-S: Scalable malware clustering based on static features," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC)*. San Jose, CA, USA: USENIX Association, Jun. 2013, pp. 187–198. [Online]. Available: <https://www.usenix.org/conference/atc13/technical-sessions/presentation/hu>
- [9] Y. Tanaka, M. Akiyama, and A. Goto, "Analysis of malware download sites by focusing on time series variation of malware," *J. Comput. Sci.*, vol. 22, pp. 301–313, Sep. 2017.
- [10] Y. Li, S. C. Sundaramurthy, A. G. Bardas, X. Ou, D. Caragea, X. Hu, and J. Jang, "Experimental study of fuzzy hashing in malware clustering analysis," in *Proc. 8th Workshop Cyber Secur. Experimentation Test (CSET)*, 2015, pp. 1–8.
- [11] Y. Li, J. Jang, X. Hu, and X. Ou, "Android malware clustering through malicious payload mining," in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*. Cham, Switzerland: Springer, 2017, pp. 192–214.
- [12] T. Chakraborty, F. Pierazzi, and V. S. Subrahmanian, "EC2: Ensemble clustering and classification for predicting Android malware families," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 2, pp. 262–277, Mar. 2020.
- [13] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLoS ONE*, vol. 11, no. 6, Jun. 2016, Art. no. e0155781.
- [14] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.
- [15] U. Fiore, F. Palmieri, A. Castiglione, and A. De Santis, "Network anomaly detection with the restricted Boltzmann machine," *Neurocomputing*, vol. 122, pp. 13–23, Dec. 2013.
- [16] Y. He, G. J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2505–2516, Sep. 2017.
- [17] A. Javid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," *Eai Endorsed Trans. Secur. Saf.*, vol. 3, no. 9, p. e2, 2016.

- [18] K. Fu, D. Cheng, Y. Tu, and L. Zhang, "Credit card fraud detection using convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process.* Cham, Switzerland: Springer, 2016, pp. 483–490.
- [19] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke, and L. Shu, "Federated deep learning for cyber security in the Internet of Things: Concepts, applications, and experimental analysis," *IEEE Access*, vol. 9, pp. 138509–138542, 2021.
- [20] M. Tavallae, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.
- [21] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, and C. Siemens, "DREBIN: Effective and explainable detection of Android malware in your pocket," in *Proc. NDSS*, vol. 14, 2014, pp. 23–26.
- [22] R. Ronen, M. Radu, C. Feuerstein, E. Yom-Tov, and M. Ahmadi, "Microsoft malware classification challenge," 2018, *arXiv:1802.10135*.
- [23] A. H. Lashkari, A. F. A. Kadir, H. Gonzalez, K. F. Mbah, and A. A. Ghorbani, "Towards a network-based framework for Android malware detection and characterization," in *Proc. 15th Annu. Conf. Privacy, Secur. Trust (PST)*, Aug. 2017, pp. 233–23309.
- [24] M. Singhal and D. Levine, "Analysis and categorization of drive-by download malware," in *Proc. 4th Int. Conf. Comput., Commun. Secur. (ICCCS)*, Oct. 2019, pp. 1–4.
- [25] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 DARPA off-line intrusion detection evaluation," *Comput. Netw.*, vol. 34, no. 4, pp. 579–595, 2000.
- [26] A. Shiravi, H. Shiravi, M. Tavallae, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, 2012.
- [27] E. B. Beigi, H. H. Jazi, N. Stakhanova, and A. A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches," in *Proc. IEEE Conf. Commun. Netw. Secur.*, Oct. 2014, pp. 247–255.
- [28] A. H. Lashkari, A. F. A. Kadir, L. Taheri, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark Android malware datasets and classification," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, 2018, pp. 1–7.
- [29] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *J. Inf. Secur. Appl.*, vol. 50, Feb. 2020, Art. no. 102419.
- [30] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-Peer Netw. Appl.*, vol. 12, no. 2, pp. 493–501, Jan. 2019.
- [31] S. MahdaviFar and A. A. Ghorbani, "Application of deep learning to cyber-security: A survey," *Neurocomputing*, vol. 347, pp. 149–176, Jun. 2019.
- [32] D. Berman, A. Buczak, J. Chavis, and C. Corbett, "A survey of deep learning methods for cyber security," *Information*, vol. 10, no. 4, p. 122, Apr. 2019.
- [33] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in IoT-edge devices," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3930–3944, Mar. 2022.
- [34] R. Zhao, Y. Yin, Y. Shi, and Z. Xue, "Intelligent intrusion detection based on federated learning aided long short-term memory," *Phys. Commun.*, vol. 42, Oct. 2020, Art. no. 101157.
- [35] Q. Qin, K. Poularakis, K. K. Leung, and L. Tassiulas, "Line-speed and scalable intrusion detection at the network edge via federated learning," in *Proc. IFIP Netw. Conf. (Networking)*, 2020, pp. 352–360.
- [36] *URLHaus Malware URL Exchange*. Accessed: Jul. 1, 2020. [Online]. Available: <https://urlhaus.abuse.ch>
- [37] *VxVault*. Accessed: Jul. 1, 2020. [Online]. Available: <http://vxvault.net>
- [38] *Virustotal*. Accessed: Jul. 1, 2020. [Online]. Available: <https://virustotal.com>
- [39] *Hybrid Analysis Free Automated Malware Analysis Service*. [Online]. Available: <https://www.hybrid-analysis.com>
- [40] *Joe Sandbox Automated Malware Analysis*. Accessed: Jul. 1, 2020. [Online]. Available: <https://joesandbox.com>
- [41] *ANY.RUN Interactive Online Malware Sandbox*. Accessed: Jul. 1, 2020. [Online]. Available: <https://any.run>
- [42] R. Dingleline, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router." Nav. Res. Lab., Washington, DC, USA, Tech. Rep., 2004. [Online]. Available: <https://apps.dtic.mil/sti/pdfs/ADA465464.pdf>
- [43] S. Zhu, J. Shi, L. Yang, B. Qin, Z. Zhang, L. Song, and G. Wang, "Measuring and modeling the label dynamics of online anti-malware engines," in *Proc. 29th USENIX Secur. Symp. (USENIX Secur.)*, 2020, pp. 2361–2378.
- [44] M. Sebastián, R. Rivera, P. Kotzias, and J. Caballero, "AVCLASS: A tool for massive malware labeling," in *Research in Attacks, Intrusions, and Defenses*, F. Monrose, M. Dacier, G. Blanc, and J. Garcia-Alfaro, Eds. Cham, Switzerland: Springer, 2016, pp. 230–253.
- [45] G. Pitolli, L. Aniello, G. Laurenza, L. Querzoni, and R. Baldoni, "Malware family identification with BIRCH clustering," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2017, pp. 1–6.
- [46] G. Pitolli, G. Laurenza, L. Aniello, L. Querzoni, and R. Baldoni, "MalFamAware: Automatic family identification and malware classification through online clustering," *Int. J. Inf. Secur.*, vol. 20, no. 3, pp. 371–386, Jun. 2021.
- [47] Y. Chen, F. Liu, Z. Shan, and G. Liang, "MalCommunity: A graph-based evaluation model for malware family clustering," in *Proc. Int. Conf. Pioneering Comput. Sci., Eng. Educators*. Singapore: Springer, 2018, pp. 279–297.
- [48] I. Shiel and S. O'Shaughnessy, "Improving file-level fuzzy hashes for malware variant classification," *Digit. Invest.*, vol. 28, pp. S88–S94, Apr. 2019.
- [49] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mashon, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the Mirai botnet," in *Proc. 26th USENIX Secur. Symp. (USENIX Secur.)*, 2017, pp. 1093–1110.
- [50] S. Sharmeen, S. Huda, J. H. Abawajy, W. N. Ismail, and M. M. Hassan, "Malware threats and detection for industrial mobile-IoT networks," *IEEE Access*, vol. 6, pp. 15941–15957, 2018.
- [51] A. Costin and J. Zaddach, "IoT malware: Comprehensive survey, analysis framework and case studies," *BlackHat USA*, 2018.
- [52] S. Ishimaru. (Apr. 13, 2017). *Old Malware Tricks to Bypass Detection in the Age of Big Data*. [Online]. Available: <https://securelist.com/old-malware-tricks-to-bypass-detection-in-the-age-of-big-data/78010/>
- [53] MITRE. (Jun. 20, 2020). *Process Injection: Process Hollowing*. [Online]. Available: <https://attack.mitre.org/techniques/T1055/012/>
- [54] T. Seals. (Dec. 2, 2019). *Authorities Break Up Imminent Monitor Spyware Organization*. [Online]. Available: <https://threatpost.com/authorities-imminent-monitor-spyware-organization/150731/>
- [55] P. Li, L. Liu, D. Gao, and M. K. Reiter, "On challenges in evaluating malware clustering," in *Recent Advances in Intrusion Detection*, S. Jha, R. Sommer, and C. Kreibich, Eds. Berlin, Germany: Springer, 2010, pp. 238–255.
- [56] J. Oliver, C. Cheng, and Y. Chen, "TLSH—A locality sensitive hash," in *Proc. 4th Cybercrime Trustworthy Comput. Workshop*, Nov. 2013, pp. 7–13.
- [57] F. Pagani, M. Dell'Amico, and D. Balzarotti, "Beyond precision and recall: Understanding uses (and Misuses) of similarity hashes in binary analysis," in *Proc. 8th ACM Conf. Data Appl. Secur. Privacy*, Mar. 2018, pp. 354–365.
- [58] A. Ahaskar. (May 2019). *Notorious Trojan Emotet Attacks Indian Firms Daily*. [Online]. Available: <https://www.livemint.com/technology/tech-news/notorious-trojan-emotet-attacks-indian-firms-daily-1558690112769.html>
- [59] P. Xu, Y. Zhang, C. Eckert, and A. Zarras, "HawkEye: Cross-platform malware detection with representation learning on graphs," in *Artificial Neural Networks and Machine Learning*, I. Farkas, P. Masulli, S. Otte, and S. Wermter, Eds. Cham, Switzerland: Springer, 2021, pp. 127–138.
- [60] Y.-T. Lee, T. Ban, T.-L. Wan, S.-M. Cheng, R. Isawa, T. Takahashi, and D. Inoue, "Cross platform IoT-malware family classification based on printable strings," in *Proc. IEEE 19th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Dec. 2020, pp. 775–784.
- [61] U. Bayer, P. M. Comparetti, C. Hlauschek, C. Kruegel, and E. Kirda, "Scalable, behavior-based malware clustering," in *Proc. NDSS*, vol. 9, 2009, pp. 8–11.
- [62] A. Mohaisen, O. Alrawi, M. Larson, and D. McPherson, "Towards a methodical evaluation of antivirus scans and labels," in *Proc. Int. Workshop Inf. Secur. Appl.* Cham, Switzerland: Springer, 2013, pp. 231–241.
- [63] A. Mohaisen and O. Alrawi, "Av-meter: An evaluation of antivirus scans and labels," in *Proc. Int. Conf. Detection Intrusions Malware, Vulnerability Assessment*. Cham, Switzerland: Springer, 2014, pp. 112–131.
- [64] (1991). *A New Virus Naming Convention*. [Online]. Available: <http://www.caro.org/articles/naming.html>

- [65] *Malware Attribute Enumeration and Characterization (MAEC)*. Accessed: Jul. 1, 2020. [Online]. Available: <http://maecproject.github.io/>
- [66] K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov, "Learning and classification of malware behavior," in *Proc. Int. Conf. Detection Intrusions Malware, Vulnerability Assessment*. Berlin, Germany: Springer, 2008, pp. 108–125.
- [67] K. Rieck, P. Trinius, C. Willems, and T. Holz, "Automatic analysis of malware behavior using machine learning," *J. Comput. Secur.*, vol. 19, no. 4, pp. 639–668, Jun. 2011.
- [68] G. E. Dahl, J. W. Stokes, L. Deng, and D. Yu, "Large-scale malware classification using random projections and neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 3422–3426.
- [69] B. Rahbarinia, M. Balduzzi, and R. Perdisci, "Real-time detection of malware downloads via large-scale URL->file->machine graph mining," in *Proc. 11th ACM Asia Conf. Comput. Commun. Secur. (ASIA CCS)*. New York, NY, USA: Association for Computing Machinery, 2016, pp. 783–794, doi: [10.1145/2897845.2897918](https://doi.org/10.1145/2897845.2897918).
- [70] A. E. El-Din, E. E.-D. Hemdan, and A. El-Sayed, "Malweb: An efficient malicious websites detection system using machine learning algorithms," in *Proc. Int. Conf. Electron. Eng. (ICEEM)*, Jul. 2021, pp. 1–6.
- [71] M. Somesha, A. R. Pais, R. S. Rao, and V. S. Rathour, "Efficient deep learning techniques for the detection of phishing websites," *Sādhanā*, vol. 45, no. 1, pp. 1–18, Dec. 2020.



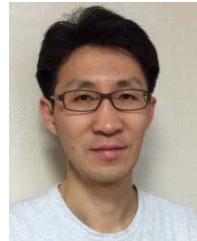
SIWON HUH received the B.S. degree in mathematics and computer science from Sungkyunkwan University, Suwon, South Korea, in 2021, where he is currently pursuing the master's degree in computer science and engineering. His research interests include blockchain identity management, malware analysis, and secure machine learning computation.



SEONGHWAN CHO received the B.S. degree in computer engineering from Sungkyunkwan University, and the M.S. degree from the Graduate School of Information Security, Korea Advanced Institute of Science and Technology (KAIST). His research interests include malware analysis and cyber threat intelligence.



JINHO CHOI received the B.S. degree from the Computer Science Department, Korea Military Academy, and the M.S. degree from the Computer Engineering Department, Texas A&M University. He is currently pursuing the Ph.D. degree with the Graduate School of Information Security, Korea Advanced Institute of Science and Technology (KAIST). His research interests include the measurement of cyber abusing with open source and cyber threat intelligence.



SEUNGWON SHIN (Member, IEEE) received the B.S. and M.S. degrees in electrical and computer engineering from Korea Advanced Institute of Science and Technology (KAIST), and the Ph.D. degree in computer engineering from the Electrical and Computer Engineering Department, Texas A&M University. He is currently an Associate Professor with the School of Electrical Engineering, KAIST. He is also the Corporate Vice President at Samsung Electronics, leading the Security

Team at the IT&Mobile Communications Division. His research interests include software-defined networking security, the IoT security, botnet analysis/detection, dark web analysis, and cyber threat intelligence.



HOJOON LEE received the B.S. degree from The University of Texas at Austin, and the Ph.D. degree from Korea Advanced Institute of Science and Technology (KAIST), in 2018, under the supervision of Prof. Brent Byunghoon Kang. He has been an Assistant Professor with the Department of Computer Science and Engineering, Sungkyunkwan University, since September 2019. Prior to his current position, he spent one year as a Postdoctoral Researcher at CISPA, under the supervision of Prof. Michael Backes. His main research interest lies in retrofitting security in computing systems against today's advanced threats. His research interests include but are not limited to operating system security, trusted execution environments, program analysis, software security, and secure AI computation in cloud.

...