Re-measuring the Label Dynamics of Online Anti-Malware Engines from Millions of Samples

Jingjing Wang
School of Computer Science, Beijing University of Posts and Telecommunications
Beijing, China
wangjingjing@bupt.edu.cn

Liu Wang*
School of Computer Science, Beijing University of Posts and Telecommunications
Beijing, China
w_liu@bupt.edu.cn

Feng Dong†
Hubei Key Laboratory of Distributed System Security, Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology
Wuhan, China
dongfeng@hust.edu.cn

Haoyu Wang†
Hubei Key Laboratory of Distributed System Security, Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology.
Wuhan, China
haoyuwang@hust.edu.cn

ABSTRACT

VirusTotal is the most widely used online scanning service in both academia and industry. However, it is known that the results returned by antivirus engines are often inconsistent and changing over time. The intrinsic dynamics of VirusTotal labeling have prompted researchers to investigate the characteristics of label dynamics for more effective use. However, they are generally limited in terms of the size and diversity of the datasets used in the measurements. This poses threats to many of their conclusions. In this paper, we perform an extraordinary large-scale study to re-measure the label dynamics of VirusTotal. Our dataset involves all the scan data in VirusTotal over a 14-month period, including over 571 million samples and 847 million reports in total. With this large dataset, we are able to revisit many issues related to the label dynamics of VirusTotal, including the prevalence of label dynamics/silence, the characteristics across file types, the impact of label dynamics on common label aggregation methods, the stabilization patterns of labels, etc. Our measurement reveals some observations that are unknown to the research community and even inconsistent with previous research. We believe that our findings could help researchers advance the understanding of the VirusTotal ecosystem.

CCS CONCEPTS

• **Security and privacy** → **Intrusion/anomaly detection and malware mitigation**; **Network security**; **Web application security**.

KEYWORDS

VirusTotal, Antivirus detection, Malware labeling, Label dynamics

ACM Reference Format:
Jingjing Wang, Liu Wang, Feng Dong, and Haoyu Wang. 2023. Re-measuring the Label Dynamics of Online Anti-Malware Engines from Millions of Samples. In *Proceedings of the 2023 ACM Internet Measurement Conference (IMC '23), October 24–26, 2023, Montreal, QC, Canada.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3618257.3624800

1 INTRODUCTION

VirusTotal [2] (VT) is the most widely used online antivirus (AV) scanning platform in both academic and industrial settings. Collaborating with over 70 security vendors, it analyzes user-submitted files using an array of security products and provides access to the scan results through free and commercial APIs or web interfaces. Users can submit a file to VirusTotal and obtain scan results from various antivirus engines. Based on the 70+ labels from different engines, users can determine the maliciousness of a scanned file, label the sample with a malware family, create their own malware datasets [6, 13, 18, 24, 37].

However, labeling samples with VirusTotal presents two common challenges. Firstly, the inconsistency of labels from different engines requires researchers to develop strategies to aggregate labels and assign a binary label (i.e., malicious or benign) to the sample. These strategies, typically defining a voting threshold, often rely on researchers' intuition and experience without a standard. Secondly, the labels returned for a given file might change over time, complicating the identification of the sample's true label.

These challenges render sample labeling via VirusTotal inherently uncertain and dynamic. Recent efforts [12, 15, 21, 26, 28, 29] have tried to analyze and measure the label dynamics of VirusTotal. They usually focus on one specific file type with a limited number

*Liu Wang is the co-first author.
†Haoyu Wang (haoyuwang@hust.edu.cn) and Feng Dong (dongfeng@hust.edu.cn) are the corresponding authors.

IMC '23, October 24–26, 2023, Montreal, QC, Canada

ACM ISBN 979-8-4007-0382-9/23/10…$15.00
https://doi.org/10.1145/3618257.3624800

of samples, such as Portable Executable (PE) files [4] [16] [27] [36], URLs [29] [34] [20], Android apps [3] [6] [14]. For example, Zhu et al. [36] performed a measurement over a year to collect daily snapshots of VirusTotal labels for a set of PE files from different antivirus engines. They revealed that label flips widely exist across files, scan dates, and engines, and they provided justifications for the most used threshold-based labeling methods (i.e., suggesting appropriate thresholds). However, their driving data is a limited dataset of their own construction with only PE files, which poses a threat to the validity of many of their conclusions. In fact, almost all existing studies have similar limitations in measurement scale, which makes it difficult for them to draw a holistic picture.

In this paper, we conduct an extensive large-scale study to re-measure the label dynamics of VirusTotal. We first collaborate with our industry partner to collect a comprehensive dataset of all samples submitted to VirusTotal and their scan results from VirusTotal's internal data for a total of 14 months. This dataset includes more than 847 million scan reports corresponding to over 571 million unique samples across 351 different file types (see **Section 4**). To the best of our knowledge, this is the largest and most extensive dataset on VirusTotal scan results in the research community. This comprehensive dataset allows us to revisit many issues of label dynamics in VirusTotal that are of interest to researchers. Specifically, we aim to explore: (*i*) What is the overall landscape of the scan data in VirusTotal, including the prevalence of label dynamics/silence, the characteristics across file types, the impact of label dynamics on label aggregation methods, etc. (*ii*) Whether the scanned results of the samples will gradually reach stability, and if so, how long a user should expect before a sample's labels become stable. (*iii*) From the perspective of individual engines, how frequently label flipping occurs in each engine, and whether there is some dependency in their detection results. Our objective is to provide an advanced understanding of VirusTotal scanning, especially with respect to the dynamic nature of VirusTotal results, to identify flawed approaches and promote better practices for scanning with VirusTotal.

First, we aim to measure the dynamics of the detection results in VirusTotal (see **Section 5**). We mainly focus on how the number of engines that detect the sample as malicious (which we call AV-Rank) varies over time for a given sample. We find an almost 50/50 split between samples with changes in AV-Rank and those that remain stable. For stable samples, most of them are benign samples (AV-Rank=0). For dynamic samples, the variation of AV-Rank is prevalent even between adjacent scans of a sample. We also find that the degree of variability in two results has a relationship with the time interval between the two scans of a sample. The results also show varying degrees of variability in terms of different file types. As for the commonly used threshold-based label aggregation method, our analysis provides justification for its ability to tolerate label dynamics and we offer suggestions for selecting thresholds.

Second, we study the stabilization of VirusTotal's detection results, for both AV-Ranks and the binary labels of samples (see **Section 6**). We find that only 10.9% of samples with multiple scans have AV-Rank finally stabilized at a constant. However, the AV-Rank of most samples tends to fluctuate in a small range. Further, for those samples where AV-Rank was able to reach stability, more than 90% achieved stability within 30 days. In addition, we set different thresholds to examine the stabilization of samples' annotation labels. We find that the annotation labels of most samples eventually become stable, mostly in 30 days (varies slightly by different thresholds). This informs the mitigation of the impact of VirusTotal dynamics.

Third, we investigate the stability and dependency of individual antivirus engines in VirusTotal (see **Section 7**). We find that the performance of individual engines differs significantly across different file types. We confirm that strong correlations exist between some engines in their labeling decisions, and we identify the groups of highly correlated engines. By analyzing these characteristics, we aim to contribute valuable information for decision-makers in selecting and optimizing antivirus engines for various tasks.

Our contributions are summarized as follows:

- We collect a thorough dataset containing all detailed data of scans from VirusTotal over a 14-month period. This is the largest VirusTotal scan dataset used for analysis in the available studies.
- We measure the overall landscape of VirusTotal scan results, including the silence and dynamics of scan results, and the impact of label dynamics on common label aggregation methods. We present some observations that are unknown to the community.
- We identify the stabilization patterns of VirusTotal scan results. This provides insights into mitigating the effects of the dynamics of VT results.
- We confirm the existence of the relationship between VirusTotal Engines' labeling decisions. We identify groups of engines with strong correlations to inform engine selection and optimization in various tasks.

2 PRELIMINARY AND RELATED WORK

2.1 VirusTotal Scanning APIs

VirusTotal provides a service interface, known as an Application Programming Interface (API), which allows users to submit and scan files/URLs and access completed scan reports. By utilizing the provided APIs, users can create simple scripts to automatically access the information generated by VirusTotal without the need for a web browser. This API facilitates a variety of functions, specifically:

- *Upload API*[1]: This API enables users to upload a file to VirusTotal using a `POST` request, allowing the file to be analyzed.
- *Report API*[2]: This API allows users to directly retrieve information about a file using a `GET` request, achieved by searching for an existing file hash.
- *Rescan API*[3]: This API ensures that users can reanalyze files already uploaded to VirusTotal by using a `POST` request to obtain rescan reports from engines without the need for re-uploading.

3 TYPES OF VT SCAN REPORTS

VirusTotal generates a scan report for each file analysis, which includes file metadata (e.g., hashes, size), certificate metadata (e.g., thumbprint), VirusTotal-specific data (e.g., submission time), and a

[1] https://www.virustotal.com/api/v3/files
[2] https://www.virustotal.com/api/v3/files/id
[3] https://www.virustotal.com/api/v3/files/id/analyse

list of detection labels assigned by over 70 antivirus engines used to scan the file. These scan reports can be generated by invoking the aforementioned VirusTotal APIs. Interestingly, we observe that the update rules for some fields in the reports differ depending on the API used to generate the report. Specifically, three fields —"last_submission_date" (the date when the file was last submitted/uploaded to VT), "last_analysis_date" (the date when the file was last analyzed), and "times_submitted" (the number of times the file was submitted/uploaded to VirusTotal) —exhibit different update rules for different APIs. To investigate this, we randomly selected several samples, called the three APIs for them multiple times, and recorded the generated reports. Through manual analysis, we summarized their update rules which correspond to three types of reports:

(*i*) *Report Generated by Upload API.* When users call the *upload API*, VirusTotal requests the user to upload the sample and then analyzes it. This process involves both submission and analysis operations, so in the reports generated by this API, "last_submission_date", "last_analysis_date" will be updated, and "times_submitted" will be increased.

(*ii*) *Report Generated by Rescan API.* For samples that already exist in VirusTotal, VirusTotal will reanalyze them when users call the *rescan API*. This process only involves analysis operation, so in the reports generated by this API, only "last_analysis_date" will be updated, and "last_submission_date", and "times_submitted" will remain unchanged.

(*iii*) *Report Generated by Report API.* When we call the report API, we are able to retrieve the latest report for a sample that has been uploaded to VT already. It is important to note that this process does not generate new reports but instead queries and returns the most recent report for the sample. As a result, there are no new reports generated, which means that fields "last_analysis_date", "last_submission_date", and "times_submitted" will not be affected, i.e., remain unchanged.

Table 1: A Brief Summary of update rules for different types of reports in VirusTotal.

	last_analysis_date	last_submission_date	times_submitted
Upload	Update	Update	Update
Rescan	Update	Unchange	Unchange
Report	Unchange	Unchange	Unchange

3.1 Using VirusTotal for Malware Labeling

VirusTotal has been widely used in academia and industry to flag malicious files, IPs, and URLs. However, due to the inconsistency and dynamics of the labels generated by VirusTotal, researchers often need to employ label aggregation methods or data processing techniques to assign a single label to the given file.

Most academic papers adopt threshold-based methods to classify files as either malicious or benign. Given a file, if the number of "malicious" labels returned by various engines exceeds or equals the threshold, the file is marked as malicious. In previous works, the threshold can be set to one [24, 30, 31, 35], two [22] or ten [30]. And some researchers use a percentage threshold, e.g., set the threshold as 50% of the engines [10, 11, 35]. Furthermore, some researchers suggest that not all engines on VirusTotal should be treated equally in terms of trustworthiness. Therefore, they often select engines with a high reputation and rely solely on reports generated by these engines [3][35] [8]. Thus, there is a lack of standards on how to assign labels to the scanned samples based on the results of the various engines. Additionally, some studies utilize the labels from VirusTotal to train their models or use them as a comparison baseline for their proposed methods [33][17]. Moreover, a subset of the research community has utilized machine learning techniques to predict the final label using the VirusTotal labeling results as input [15] [28].

3.2 Dynamics of VirusTotal

The dynamics of VirusTotal have been a topic of interest for researchers in the cybersecurity field [19, 20, 28, 32, 36]. For instance, Peng et al. [19] examined the labeling process of third-party vendors of VirusTotal focusing on phishing URLs over one month. Choo et al. [7] conducted an in-depth analysis of VirusTotal's URL scanning reports, uncovering challenges in determining URL maliciousness due to conflicting scanner outputs. Peng et al. [20] investigated VirusTotal's URL scan API, highlighting the platform's blackbox nature in detecting phishing websites and the need for a deeper understanding of its labeling mechanisms. Zhu et al. [36] conducted an experiment over one year to measure the label dynamics of VirusTotal by collecting daily snapshots for their dataset (PE files). Wang et al. [32] explored the dynamics and uncertainty of VirusTotal's labeling on Android malware, and revealed that relying on VirusTotal for malware labeling is prone to generating error/noisy labels. However, these studies focused on limited types of samples, such as URLs or PE files, and faced either scale or temporal limitations. Measuring only a small number of samples is not representative of the overall picture on VirusTotal, and the short measurement time does not give a complete view of dynamic trends.

In general, the label dynamics of VirusTotal is an important and complex issue. Our work is an incremental step in this line of research. In contrast to these studies, our research stands out for its broader, more exhaustive dataset and more systematic, in-depth analysis. We revisit many of the aspects mentioned in related work with our own unique insights. More importantly, we venture into areas previously untouched, such as file type-specific analysis, and stable sample-specific examination. This comprehensive approach ensures that our findings not only build upon but significantly expand the existing body of knowledge.

4 DATASET

4.1 Data Collection

Data source. To comprehensively measure and analyze VirusTotal's labeling results, we need to harvest a comprehensive dataset of VT scan results. However, compiling such a dataset requires considerable effort to continuously submit a large number of samples to VirusTotal and obtain scan results. To address this challenge, we collaborated with a leading security company to purchase the premium license of VirusTotal, which enabled us to collect large-scale internal data on the VirusTotal service. Specifically, VirusTotal provides a special interface for users who purchase the premium license to access their data. We called this interface every minute

and VirusTotal returned us all the scan reports generated in that minute. We cached and parsed the scan reports, compressed them, and stored them in a MongoDB database. Our data collection process lasted for 14 months from May 2021 to June 2022, and thus we obtained VT scan data for all samples submitted to VT during this period. In total, we collected 847,567,045 scan reports corresponding to 571,120,263 unique samples (by hash) covering 351 different file types[4]. Of these samples, 524,079,708 (91.76%) are "fresh" samples, i.e., the samples were submitted to VirusTotal for the first time within the data collection period.

Data pre-processing. To efficiently process the huge amount of data collected from VirusTotal, we utilized a MongoDB database and data compression techniques. The sample's basic information and scan results are stored separately to reduce data redundancy, and only the relevant fields are stored, which achieves a compression rate of 10.06 times. This approach allows for efficient analysis of the dataset. Table 2 presents the overall statistics of the collected dataset.

Table 2: Overview of our dataset (stored by month).

Month	Count	Size
05/2021 Reports	41,336,308	38.309 GB
06/2021 Reports	51,945,339	47.358 GB
07/2021 Reports	59,538,559	52.064 GB
08/2021 Reports	60,369,255	53.780 GB
09/2021 Reports	64,546,564	56.892 GB
10/2021 Reports	55,113,116	51.701 GB
11/2021 Reports	57,728,868	53.306 GB
12/2021 Reports	59,421,199	53.740 GB
01/2022 Reports	69,676,958	63.630 GB
02/2022 Reports	61,981,425	55.816 GB
03/2022 Reports	76,759,558	70.044 GB
04/2022 Reports	68,555,398	55.137 GB
05/2022 Reports	62,400,644	50.993 GB
06/2022 Reports	58,193,854	50.668 GB
Total # Reports	847,567,045	753.438GB
Total # Samples	571,120,263	-

4.2 Data Distribution

4.2.1 File Type Distribution. We first examine the distribution of file types in all collected samples, which amounts to a total of 351 different file types. Notably, we observe that the top 10 file types make up 78.17% of the total samples, and the top 20 file types account for 87.04% (excluding `NULL` file type). Table 3 provides a breakdown of the number and percentage of samples and reports for the top 20 file types. The most common file type in VirusTotal scanning is `Win32 EXE`, which accounts for a quarter of the total samples in our collected dataset.

[4]The file types are defined by VirusTotal. In each VT scan report there is a field indicating the type of the scanned sample.

Table 3: The distribution of file types in our dataset.

File Type	# Samples	% Samples	# Reports	% Reports
Win32 EXE	144,001,742	25.2139%	246,581,459	29.0929%
TXT	73,547,380	12.8777%	95,522,515	11.2702%
HTML	55,741,601	9.7600%	65,727,729	7.7549%
ZIP	31,639,183	5.5398%	83,639,389	9.8682%
PDF	22,553,180	3.9489%	39,336,967	4.6412%
XML	22,038,921	3.8589%	23,794,254	2.8074%
Win32 DLL	15,857,880	2.7766%	63,214,035	7.4583%
JSON	14,440,396	2.5284%	17,748,358	2.0940%
DEX	12,761,860	2.2345%	17,596,789	2.0762%
ELF executable	11,003,126	1.9266%	12,583,631	1.4847%
Win64 EXE	8,297,574	1.4529%	28,785,071	3.3962%
Win64 DLL	6,784,214	1.1879%	17,529,977	2.0683%
ELF shared library	5,790,378	1.0139%	6,504,678	0.7675%
EPUB	5,292,965	0.9268%	9,023,941	1.0647%
LNK	4,918,415	0.8612%	5,636,696	0.6650%
FPX	4,365,032	0.7643%	5,662,784	0.6681%
PHP	3,974,229	0.6959%	4,286,022	0.5057%
DOCX	2,165,828	0.3792%	3,474,592	0.4099%
GZIP	2,164,495	0.3790%	3,455,231	0.4077%
JPEG	2,025,482	0.3547%	2,812,225	0.3318%
NULL	54,855,111	9.6048%	61,194,945	7.2201%
Others	66,901,271	11.7140%	33,455,757	3.9473%
Total	571,120,263	100.0000%	847,567,045	100.0000%

4.2.2 The Number of Scan Reports per Sample. It can be observed that the number of scans obtained by each sample was unequal. We aggregate the reports by their associated samples (i.e., sample hashes), and the distribution of the number of reports per sample is plotted. As shown in Figure 1, nearly 99.90% of the samples have less than 20 scan reports, 99.10% of samples have less than 6 reports, and 88.81% of samples have only one report. Only 565,471 samples (less than 0.1%) have more than 20 scan reports, and the largest number of reports for a single sample reaches 64,168. Overall, there are 63,999,984 samples that have more than one scan report, which is the focus of our study due to their potential to study the dynamic characteristics of VT.

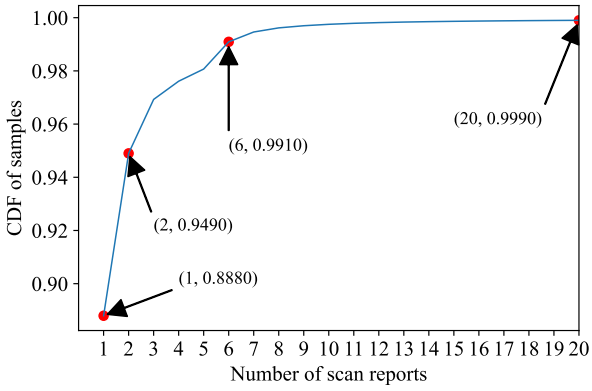


Figure 1: CDF of the number of reports per sample.

5 MEASUREMENT OF VT DYNAMICS

Given a sample, VirusTotal aggregates the detected labels from multiple antivirus engines and reports the number of engines that identify the sample as "malicious", which is called *AV-Rank*. This is reflected in the "positive" field of the VirusTotal scan reports. In general, AV-Rank can serve as an indicator to assess the level of *maliciousness* of a sample [5], i.e., the larger the AV-rank, the greater the maliciousness. As aforementioned, most existing studies rely on a threshold-based approach to determine the label of a sample, where a voting threshold is set and a sample is classified as malicious if its AV-Rank exceeds (or equals) the threshold. Consequently, fluctuations in AV-Rank of a sample can have a significant impact on its label determination. In this section, we explore the changes in AV-Rank for a large-scale sample set to gain a macroscopic perspective of the dynamics of VirusTotal results.

5.1 Stable Samples VS. Dynamic Samples

It is widely recognized that VirusTotal scan results vary over time. However, the precise landscape of this variation remains unclear, such as the prevalence of samples with changing results and the number of samples with stable results over time. To gain a better understanding of this landscape, we analyzed the development of AV-Ranks for the scanned samples. Specifically, we focused on the 63,999,984 samples (11.21%) with multiple reports, as their AV-Rank dynamics were measurable (the evolutionary trajectory of the AV-Ranks could not be captured for the sample with only one report). We divided the samples into two categories: (*i*) stable samples, which had a constant AV-Rank over multiple scans, and (*ii*) dynamic samples, which exhibited fluctuations in their AV-Ranks over time. To differentiate between these two categories, we calculated the difference between the maximum and minimum values of *positives* for each sample, denoted as $\Delta_i = p_{max} - p_{min}$. If $\Delta_i = 0$, we classified the sample as a stable sample; otherwise, it was classified as a dynamic sample. Our analysis revealed that 31,933,112 samples (49.90%) belonged to the stable samples, and 32,066,872 samples (50.10%) were categorized as dynamic samples. This indicates that, overall, there is almost a 50/50 split between samples that have changing scan results and those that remain stable.

Further, to explore whether there is bias due to an imbalance in the number of reports (e.g., stable samples may be driven by fewer reports), we compared the distribution of the number of reports between stable and dynamic samples. As shown in Figure 2, we can observe a striking similarity in the distribution of the number of reports for stable and dynamic samples. 67.09% of the stable samples and 71.3% of the dynamic samples have only two scan reports. For both categories of samples, approximately 94% of the samples had no more than 4 reports, and 99.9% of the samples had no more than 20 reports. Since the distribution of the number of reports between stable and dynamic samples is closely aligned, we consider that the above conclusion is not subject to bias from the unbalanced number of reports in the two sample categories.

Observation 1: *In summary, for over 63 million samples with multiple reports, roughly half of the samples (50.1%) exhibit fluctuations in their AV-Ranks over time, while the other half (49.9%) have constant AV-Ranks.*

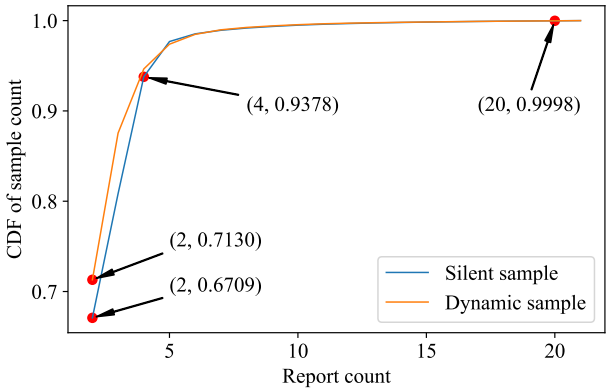


Figure 2: CDF of the number of reports for stable and dynamic samples.

5.2 Characterizing Stable Samples

Stable samples have a constant AV-Rank over multiple scans during our collection period. To better understand the characteristics of these stable samples, we analyze the distribution of their AV-Rank and the time span over which they remain stable.

5.2.1 AV-Rank Distribution. We first take a look at which number the AV-Ranks are fixed on for the stable samples, which indicates their maliciousness. As shown in Figure 3, the AV-ranks range from 0 to 69. We can observe that the majority (66.36%) of stable samples have a constant AV-rank of 0, indicating that they are basically benign samples. Over 80% of stable samples have AV-Ranks no more than 5. Although there are some samples with large AV-Ranks that remain stable, we find that they were mostly scanned fewer times. About 82% of the samples with AV-Rank > 0 were scanned only twice with an average of 2.92 scans, while 59% of the samples with AV-Rank = 0 were scanned only 2 times with an average of 3.54 scans. Thus if we ignore the samples with only 2 scans to prevent randomness, there are 81.7% of the stable samples can be considered benign samples.

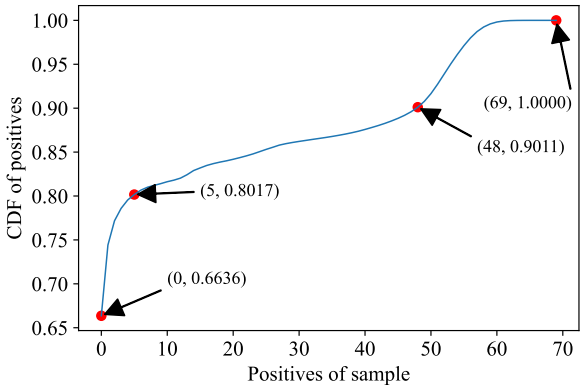


Figure 3: CDF of AV-Ranks of stable samples.

5.2.2 AV-Rank vs. Time Span of Scans. To analyze the duration for which stable samples remain stable in AV-Rank, the time interval between the last and the first scan is calculated. Overall, we found that half of the samples have a time interval within 17 days, and over 93% of samples within 350 days. This suggests that most of the stable samples maintained a table state for a limited period of time. We then group these time intervals by AV-Rank, as shown in Figure 4, where the outlier examples are excluded for conciseness. The orange lines in the box indicate the median, while the green triangles represent the mean. We can see that the benign samples (AV-Rank = 0) had the longest stable time span, with an average of 20.34 days and a median of 14 days. This suggests that benign samples have more potential to remain stable results in VirusTotal scanning.

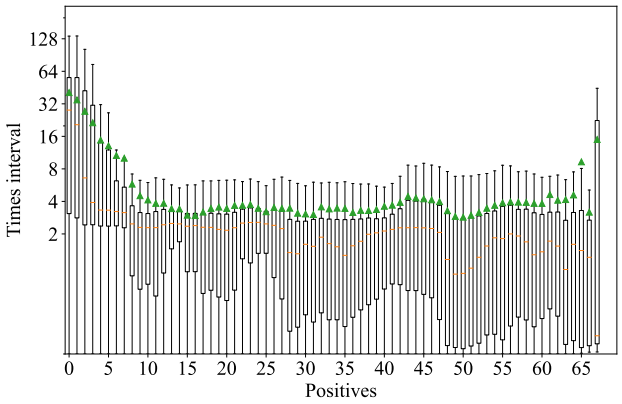


Figure 4: Distribution of time intervals under different AV-Rank for stable samples.

Observation 2: *Most (over 66.36%) of stable samples are benign samples. During our 14-month collection, approximately half of the stable samples with a stable AV-Rank within 17 days. Benign samples have more potential to maintain stable results in VirusTotal scanning.*

5.3 A Deep Dive into the Fresh Dynamic Samples

We next turn direction to dynamic samples that have changes in AV-Rank for multiple scans (i.e., $\Delta_i > 0$).

5.3.1 Construction of fresh dynamic dataset. For more robust analysis, we first pick out a main sub-dataset (denoted as S) following two principles: (*i*) The samples must be newly submitted to VirusTotal within the 14-month period for the first time so that we can observe the label dynamics from the very beginning without being distracted by the files' previous history. (*ii*) The samples belong to the top 20 file types with the largest number in VirusTotal scan data. There are a total of 351 file types identified; however, the top 20 file types account for the vast majority of the total samples. We focus our analysis on these top 20 file types, which ensures a well-organized study dataset without losing valuable insights and understanding of most types of files in real-world situations. As a result, the dataset S contains 32,051,433 samples and encompasses 109,142,027 scan reports. This set of samples is essential for examining the dynamic characteristics of labels in VirusTotal scans.

5.3.2 Dynamics Measurement Metrics. For each sample in S, we sort its scan reports by time and calculate the difference in the AV-Rank between adjacent reports: $\delta_i = |p_i - p_{i-1}|$ $(2 \leq i < n)$ where p_i represents the AV-Rank value of the i-th report and n is the total number of reports for the sample. We also calculate $\Delta_i = |p_{max} - p_{min}|$, the difference between the maximum and minimum AV-Rank values. We then analyze the distributions of these difference values (δ_i and Δ_i) across different file types and time intervals.

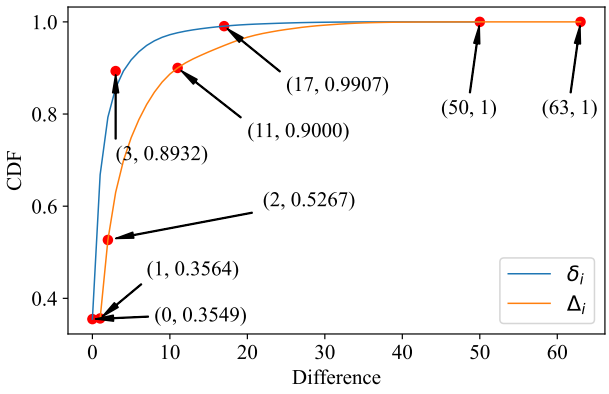
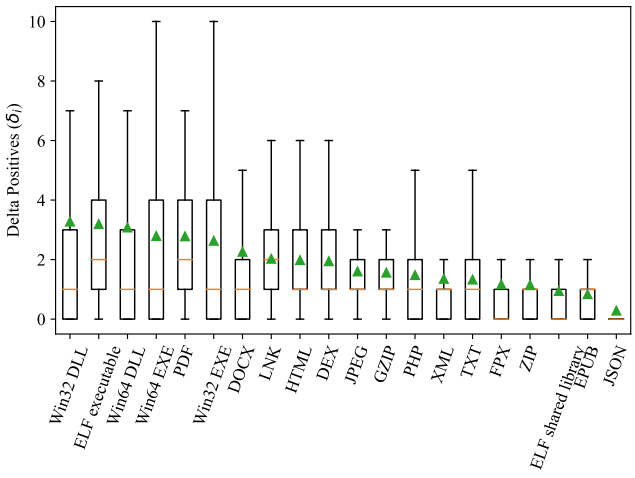


Figure 5: CDF distribution of δ_i and Δ_i.

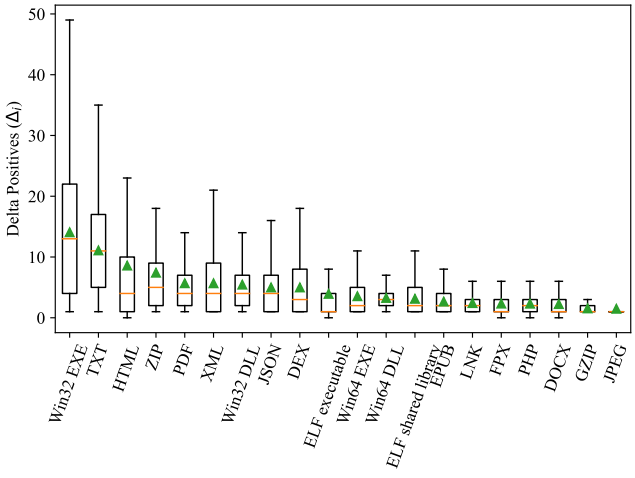
5.3.3 Overall Distribution. Figure 5 shows the CDF distribution of δ_i and Δ_i for samples in S. We can see that the value of δ_i ranges from 0 to 50, with 35.49% of the cases having δ_i equal to 0, indicating that most of the cases are two adjacent scans in which the AV-Ranks changed. Thus, the variation in AV-Rank is prevalent even in adjacent scans. For Δ_i, the differences in samples' AV-Ranks are slightly more pronounced, with the range from 1 to 63. Roughly half of the samples have Δ_i greater than 2 and 90% of samples have Δ_i within 11. There are still some samples that have larger Δ_i, showing significant changes. Overall, Δ_i predominantly falls within a range of 1-17.

Observation 3: *Overall, variation in detection results between adjacent scans of a sample is prevalent, with 65% of cases showing changes. When considering multiple scans, the differences in samples' AV-Ranks are slightly more pronounced, which concentrate in a range of 1-17.*

5.3.4 Distribution Across File Types. For each file type, we aggregate the difference values (δ_i and Δ_i) for all the samples in that file type, and depict the boxplot distribution as shown in Figure 6. The orange lines in the box indicate the median, while the green triangles represent the mean. For the sake of conciseness, we exclude the outlier examples in the figures. We can see that δ_i and Δ_i are unevenly distributed among the various file types. From Figure 6(a), the `Win32 DLL` file type has the highest δ_i between two scans, with a mean of 3.25 and a median of 1.0. In contrast, the file type with



(a) Distribution of δ_i



(b) Distribution of Δ_i

Figure 6: Distribution of δ_i and Δ_i under different file types.

the lowest average δ_i overall is `JSON`, with a mean of 0.29 and a median of 0. In Figure 6(b), the average Δ_i values for different file types range from 1.49 (`JPEG`) to 14.08 (`Win32 EXE`). These findings indicate that the variation of AV-Rank presented in different file types is uneven, and researchers could give special attention to certain file types (e.g., `DLL`, `EXE`). Besides, we find that both δ_i and Δ_i maintain low dynamics in several file types, including `EPUB, FPX, JPEG, ELF shared library, GZIP, PHP`, etc. For some file types (e.g., `ZIP, JSON, TXT`), although the differences between adjacent scan reports are small, the overall dynamics are high. This indicates that the stability of the sample cannot be determined solely based on the difference between adjacent reports.

Observation 4: *In VirusTotal, samples of different file types show varying degrees of variability in their detection results. Some file types such as* `Dll, EXE` *have higher dynamics requiring extra attention. Although most adjacent scans show small differences, overall dynamics should be considered for an accurate assessment of sample stability.*

5.3.5 Distribution Across Time Intervals. Intuitively, after a sample has been scanned once, it is more likely to yield a close result when it is scanned again a day apart versus a long time later. We next investigate whether the time difference between the two reports has some effect on the degree of their AV-Rank variation. For any two scans of each sample, we calculate their difference in AV-Rank as well as the difference in time intervals. Figure 7 shows the boxplot with the time intervals (in days) as the horizontal axis and the difference values of any two reports as the vertical axis. The green triangles in the box represent the mean value, while the orange lines represent the median value. The minimum time interval is less than one day, while the maximum is up to 418 days. We can observe that generally, the differences between two scan results tend to increase as the time intervals increase. For statistical evidence, we calculate the Spearman correlation coefficient between the difference values and time intervals. As a result, the correlation is as high as 0.9181, indicating a strong positive correlation. The p-value, which is 2.6083e-167, is extremely small, suggesting that the correlation is statistically significant. This implies that there is a strong correlation between the different values and time intervals of the two scans.

Observation 5: *The time interval between two scans of a sample has an effect on the degree of variance in the results, i.e., the difference of the positive field tends to be greater when the time interval is longer.*

5.4 Impact of AV-Rank Dynamics

As aforementioned, most researchers only submit a sample to VirusTotal once and use a voting threshold t to aggregate the results. If the number of positive engines (i.e. the AV-Rank) exceeds or equals t, the scanned sample will be assigned as malicious. However, as the AV-Rank of a sample is often changing over time, the aggregated label is likely to shift with temporality. Prior study [36] has estimated the potential impact of result variation on this label aggregation strategy for different t. It revealed that the engine's decision flipping can heavily influence the label aggregation results when t is too small or too large, and suggested that choosing t from the range of 2-39 is likely to make the aggregated labels stable. However, given the limitations of their underlying dataset (insufficient number and only PE files), we are interested in revisiting this issue following their measuring method.

5.4.1 Categorization of Samples. Based on the samples' AV-Ranks and a threshold t, we can classify the samples in our dataset into three categories: *white samples*, *black samples*, and *gray samples*. Given a threshold t, a sample with $p_{max} \leq t$ (i.e., all the AV-Ranks of the sample are less than t) is considered a white sample; a sample with $p_{min} \geq t$ (i.e., all the AV-Ranks are greater than or equal to t) is considered a black sample; otherwise, it is considered a gray sample. Clearly, the gray samples have the potential to get inconsistent labels when labeled at different times. We thus measure the proportion of gray samples under varying thresholds. Note that we focus only on the fresh dynamic samples (i.e., the dataset S) here since the stable samples are always labeled consistently with a given threshold and thus have no impact on this threshold-based labeling strategy.

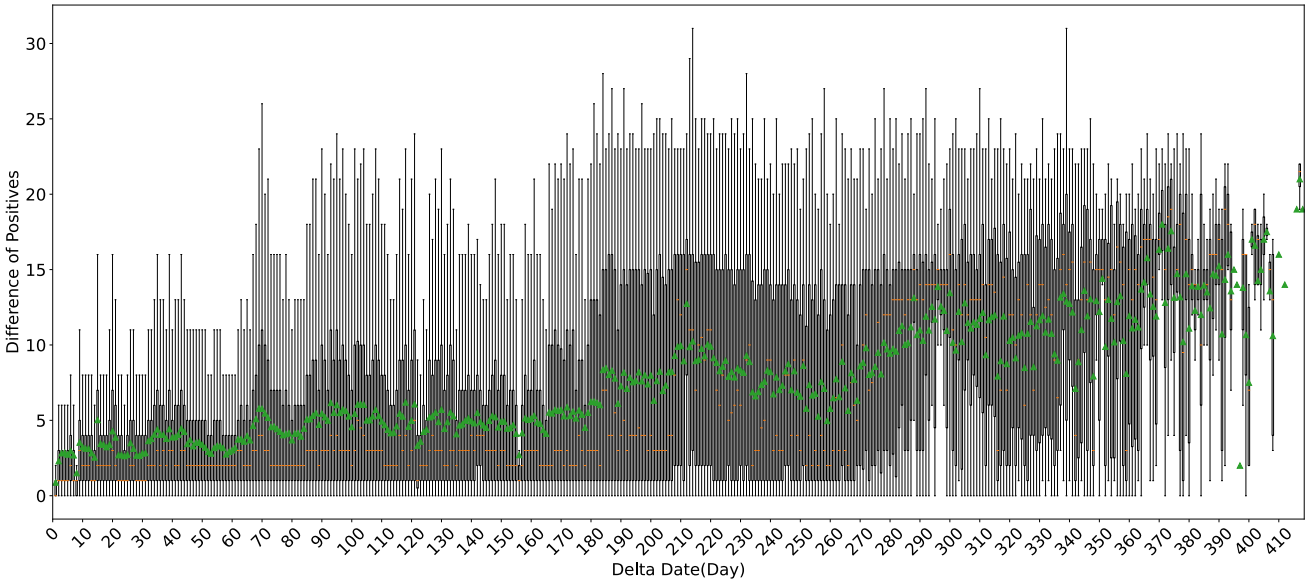


Figure 7: Distribution of differences of AV-Ranks under different time intervals.

5.4.2 Overall Distribution. We start with analyzing all the available data in S, which contains over 32 million samples with over 109 million scan results. Figure 8(a) presents the percentage distribution of the three categories of samples when t is set from 1 to 50 in increments of 1. The maximum proportion of gray samples is 14.92% when $t = 24$, and the minimum proportion is 3.82% when $t = 45$. This basically provides justification for the threshold-based labeling method that it has the potential to tolerate the label dynamics of VirusTotal. Broadly speaking, we observe that as t increases, the proportion of gray samples initially increases and then decreases. When t is in the range from 1 to 11 as well as 28 to 50, the proportions of gray samples are all lower than 10%, ranging from 3.82% to 9.41%. This contradicts previous research [36], which suggested that the proportion of gray samples first decreases and then increases as the threshold increases.

5.4.3 Distribution for PE File Type. Since the previous study [36] was conducted on PE files which suggested a reasonable range of 2-39 for threshold selection, we also conducted an exclusive investigation on PE files in our dataset. Moreover, previous work [1] [25] has suggested that PE files are the most popular submitted file type on VirusTotal, and our dataset validates this (see Table 3). Specifically, we incorporate the file types including `Win32 EXE`, `Win32 DLL`, `Win64 EXE`, `Win64 DLL` as PE files. Figure8(b) shows the distribution of three categories of samples for PE files. The maximum proportion of gray samples is 16.41% when $t = 50$, while the minimum proportion of gray samples is 2.70% when $t = 3$. Broadly speaking, we can see that the proportion of gray samples gradually increases as the threshold grows. When t is within 24, the proportion of gray samples remains less than 10% (between 2.70% and 9.85%). Thus, for the PE file type, we would recommend an appropriate selection of threshold t ranging from 1 to 24.

Observation 6: *The overall picture suggests that when selecting t from the range of 1 to 11 and 28 to 50, the aggregated labels are likely to be stable. Particularly, we find that for PE files the sensible range of t is 1 to 24. Our findings are somewhat discrepant from the conclusion of prior work [36].*

5.5 Inferring Causes of Label Dynamics

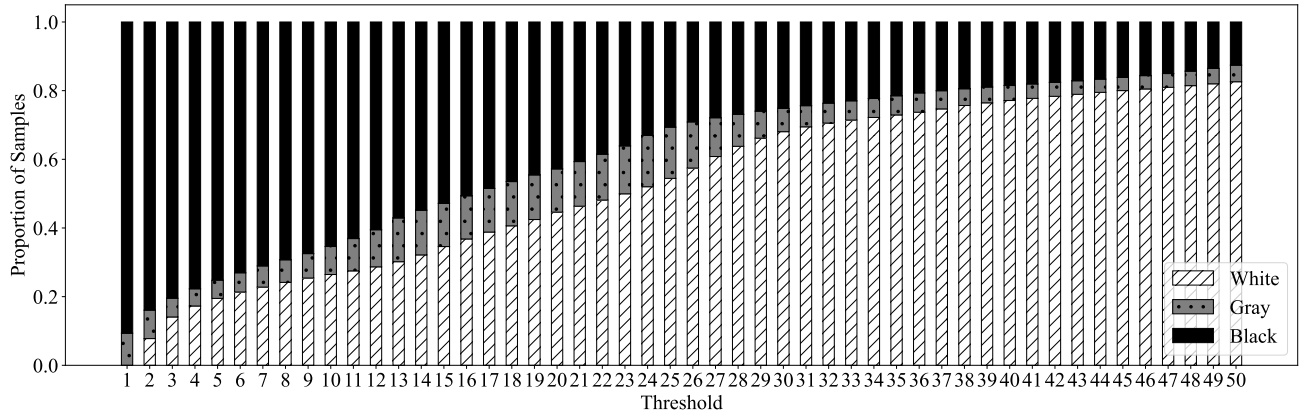
Next, we try to infer the possible causes for VT label dynamics, which are categorized as follows.

(*i*) *Engine Latency.* Changes in labels may be caused by the latency of engine results. For instance, an engine may not be able to detect a malicious sample at first due to factors such as limited knowledge or insufficient training. However, as the sample gains more circulation and its behavior gets more visibility, the previously ineffective engines may eventually update their detection capabilities and change the label. This change in label can be considered a learning process for the engine.

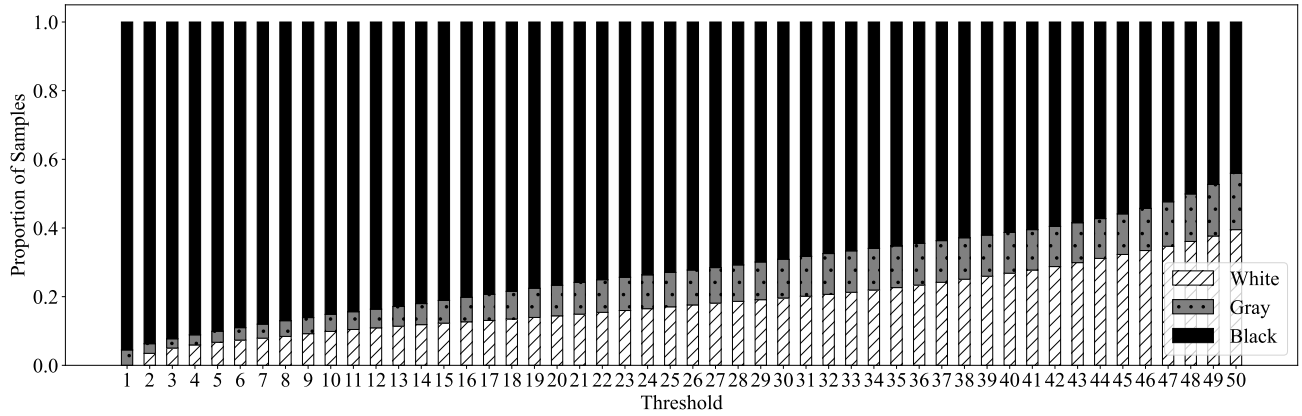
(*ii*) *Engine Update.* Previous work [36] indicated that engines' model update is the major reason for label changes. For this, we check whether the engine update occurs when two consecutive results of the engine change. The results show that engine updates are present in about 60% of the flips in our dataset. Therefore, we believe that engine update is one of the causes of label changes, as [36].

(*iii*) *Engine Activity.* There is a common situation where the activity of the engines in VirusTotal scanning is not always the same. The active engines in VirusTotal are variable, which also leads to variations in the results of sample scans. For example, in the first scan of a sample, a set of engines gave their respective labels, but some of these engines failed to detect (due to timeouts, etc.) in the second scan, which led to different results between the two scans of a sample. However, if these "inactive" engines give valid results, they are usually consistent with the original results.

Observation 7: *We identify three causes of label changes in VirusTotal scanning, including engine latency, engine update, and engine*



(a) for the overall main dataset S



(b) for the PE files in S

Figure 8: Distribution of three sample categories under different thresholds.

activity. Understanding these causes can aid in improving the practice of VirusTotal labeling.

6 LABEL STABILIZATION

So far, it has been observed that the VT scan results may vary over time. The subsequent investigation focuses on whether the scan results will eventually achieve stability after a certain period of time and, if so, how much time it would typically take. Specifically, we analyze the stabilization patterns of the AV-Rank and the binary labels for the samples in dataset S.

6.1 Stabilization of AV-Rank

We first observe the temporal evolution of AV-Rank. In simple terms, the AV-Rank reaches stability, meaning that it stops changing after a certain period of time. Based on this criterion, we examine the samples in the dataset whose AV-Rank finally remains unchanged. As a result, for all samples with more than one scan, 10.9% of them had AV-Rank that finally stabilized at a constant. This indicates that there are only a small number of samples for which the AV-Rank can finally reach constant. Further, among these samples whose AV-Rank can reach stability, we find that 90.38% of them reached stability within 30 days, 85.35% reached stability within 20 days, and 78.16% within 10 days.

In practice, sometimes the number of detection engines changes or the engine results return exceptionally (e.g., due to timeouts), so we consider it rather harsh to regard AV-Rank as stable when it remains completely constant. In this regard, we refer to a sample as reaching stability if its AV-Rank eventually stabilizes within a small fluctuation range. To this end, we define a fluctuation range, denoted as $R = \{1, 2, 3, 4, 5\}$, and a sample was said to have reached stability if its AV-Rank changed only within $r(r \in R)$ from a certain time. When the range of AV-Rank changes within 1 (r=1) is considered as sample stability, there are 55.1% of the samples that can reach stability, much higher than r=0 (10.9%). Furthermore, with a fluctuation range of 2, 3, 4, and 5, the percentage of samples that can reach stability is 69.58%, 77.84%, 83.52%, and 88.11%, respectively. The results show that although a small number of samples eventually remained constant, the AV-Rank of most samples tended to fluctuate in a small range. In addition, for samples that can reach stability under a fluctuation range of 1-5, respectively, 90.36% (1), 92.53% (2), 93.92% (3), 94.94% (4), 95.68% (5) of them reached stability within 30 days.

Observation 8: *The AV-Rank of a very small fraction of the samples (10.9%) ended up being constant, but the AV-Rank of most of the samples tended to fluctuate in a small range. Particularly, for the samples where AV-Rank was able to reach stability, more than 90% reached stability within 30 days.*

6.2 Stabilization of File Labels

We have explored the stabilization of AV-Ranks, ultimately, we are more concerned with how long a user should before a sample's labels (malicious or benign) become stable.

We still measure the stabilization of sample labels in the cases of different thresholds ($t = \{2, 5, 10, 15, 20, 25, 30, 35, 40\}$). For each sample, we model the label dynamics in the form of a sequence of "B" and "M". Specifically, given a sample s and one of its AV-Rank p_i, if the $p_i \geq t$, the sample is regarded as malicious (coded as "M"). Otherwise, we regard the sample as benign (coded as "B"). Thus, the label we obtained can be formulated as $C_s = [c_1, c_2, ..., c_t, ..., c_n]$, where n is the total number of reports of s and c_i="B" (benign) or "M" (malicious). Next, we search the label sequence to see if there is a moment from which all the labels no longer change, i.e., followed by all "M" or "B".

Figure 9 displays the results on the stability of file labels under different thresholds. The horizontal axis represents the threshold t which defines the file label as either malicious or benign. The vertical axis on the left represents the serial number of scans for the file to reach stability, and the vertical axis on the right represents the average number of days for the file to reach stability. When all samples in S are considered, the results are shown in Figure 9(a). We can see that under different thresholds, the samples begin to stabilize in the second to third reports on average. And the average time for stability is between 9.4 ($t = 40$) and 10.6 days ($t = 15$).

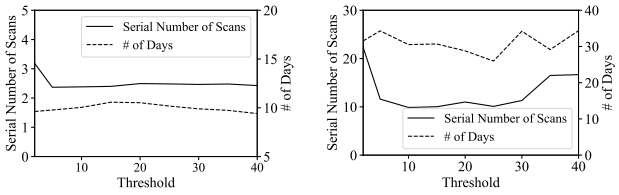
Considering that a large number of samples in S are only submitted to VT twice, which makes stability easy to achieve, we then exclude the sample with only two scans, and the results are shown in Figure 9(b). File labels are mostly stable in the 10th and 11th scans. And the maximum period for stability is more than 34 days, when $t = 40$, and the minimum is about 26 days when $t = 25$.

Under the threshold range of t = {2, 5, 10, 15, 20, 25, 30, 35, 40}, a significant majority of the samples, ranging from 93.14% (at $t = 25$) to 98.04% (at $t = 40$), have been found to reach stability. And 87.02% ($t = 20$) - 88.32% ($t = 30$) of file labels become stable after 15 days. While after 30 days, the proportion of stable samples increases to 91.09% ($t = 20$) - 92.31% ($t = 30$). This has important implications for relevant researchers in their studies to mitigate the impact of VirusTotal dynamics.

Observation 9: *Under different thresholds, the vast majority (93.14%-98.04%) of the file labels can eventually reach stability, and 91.09% - 92.31% of file labels become stable after 30 days.*

7 DETECTION OF INDIVIDUAL ENGINES

In this section, we outline the level of instability of the engines and the correlations between different engines' labeling decisions. This knowledge can assist researchers in choosing the appropriate aggregation method, based on specific engines when utilizing VirusTotal.



(a) For all the samples (b) Excluding samples with two scans

Figure 9: Stabilization of file labels.

7.1 Engine Stability

7.1.1 Label Flips. Given a sample s and a VirusTotal engine e, the label we obtained from e for s can be formulated as $S_{e,s} = [l_1, l_2, ..., l_t, ..., l_n]$ where n is the total number of scan reports of s, and $l_t = 0$ (benign) or 1 (malicious). In this context, we refer to a change between two consecutive labels, i.e., "$1 \rightarrow 0$ " or "$0 \rightarrow 1$" as a label flip. Then, we enumerate 109 million reports of 32 million samples that can be used to study the dynamic characteristics, and find that there are 16,838,818 label flips in total (1 flip per report on average), including 12,270,147 times from 0 to 1, and 4,568,671 times from 1 to 0. Previous work [36] defines "$0 \rightarrow 1 \rightarrow 0$" and "$1 \rightarrow 0 \rightarrow 1$" as "hazard flip", and finds that more than half of the flips on VirusTotal are hazard flips. However, in our dataset, we find that there are only 9 hazard flips in total (3 "$0 \rightarrow 1 \rightarrow 0$" and 6 "$1 \rightarrow 0 \rightarrow 1$ "). Thus, contrary to the conclusion mentioned in [36], "hazard flips" are not common in real-world data of VirusTotal scanning. We speculate that the reason may be because they are rescanning the sample once a day, while we are using the real data from VT, which is not scanned as frequently as compared to them, and therefore they have a stronger degree of label flips than we do.

7.1.2 Percentage of Flips Across Engines and File Types. We next examine the behaviors of different VirusTotal engines for different file types. As shown in Figure 10, we calculate the flipping ratio for each engine on each type of sample. This flipping ratio represents how often label flips occur by an engine in scanning a type of sample (i.e., $1 \rightarrow 0$, or $0 \rightarrow 1$). We can see that the performance of each engine can vary significantly depending on the file type being examined. For instance, *Arcabit* showed a higher flip ratio when dealing with `ELF executable` files (25.78%) compared to `DEX` files (0.05%). Such variations in engine performance should be considered by engine managers for targeted improvements of models, and by users for adjusting their level of trust in the engines. Besides, we observed that overall, some engines (e.g., *Arcabit, F-Secure, Linoic*) were more prone to flipping their detection results, while some (e.g., *Jiangmin, AhnLab*) exhibited more stable performance. Interestingly, even some well-known and reputable engines like *F-Secure* and *Microsoft* showed a significant number of flips. These findings underscore the need for continuous evaluation and improvement of antivirus engines to ensure optimal performance across all file types.

Observation 10: *The performance of antivirus engines varies for different file types. This should be aware of by engine managers and users to promote better VT scanning practices.*

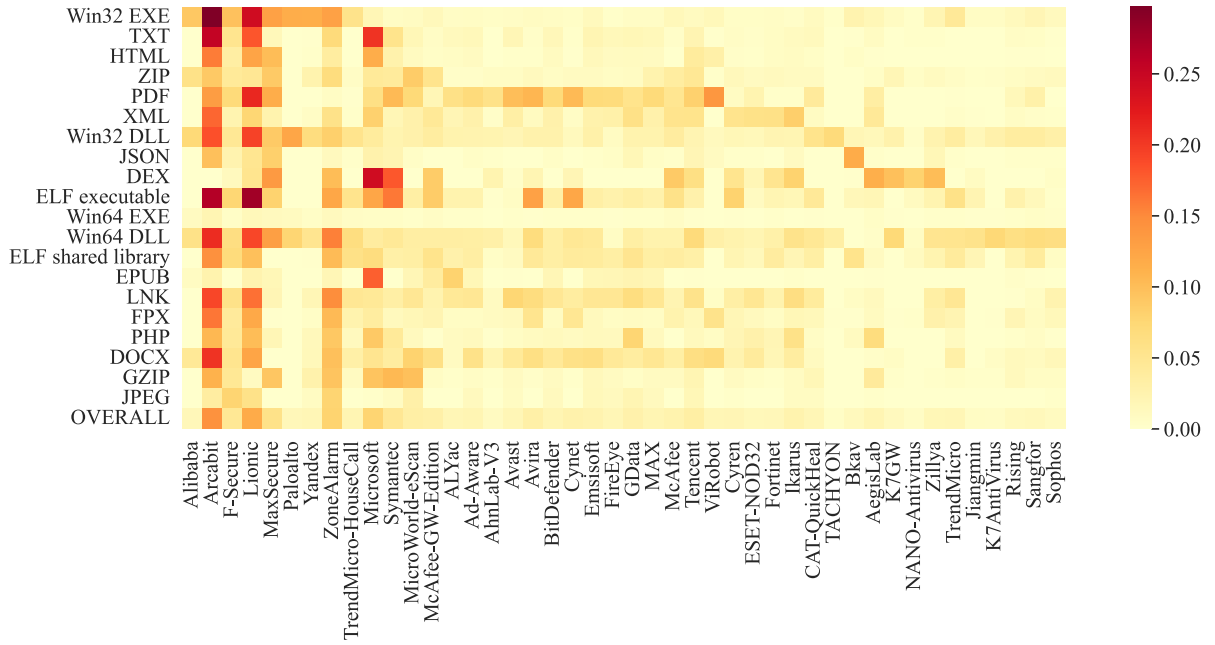


Figure 10: Flip ratio of Individual Engines across engines and file types.

7.2 Engine Correlation

Previous studies [7, 23, 36] have raised the concern about the dependency between engines. For example, Sebastián et al. [23] posed that groups of AV vendors often copy labels from each other. This motivated us to measure the correlations between different engines' labeling decisions.

As aforementioned, each AV engine reports its scan results indicating whether the scanned sample is malicious, benign, or undetected. Formally, let $S = \{s_1, s_2, \cdots, s_n\}$ be the set of n samples. Suppose there are m engines in VirusTotal, i.e., $E = \{e_1, e_2, \cdots, e_m\}$. We represent the scan results of samples using a matrix $R = \{R_{ij} | 1 \leq i \leq n, 1 \leq j \leq m\}$ where

$$R_{ij} = \begin{cases} 1, & \text{if } s_i \text{ is detected as malicious by } e_j \\ 0, & \text{if } s_i \text{ is detected as benign by } e_j \\ -1, & \text{if } s_i \text{ is undetected by } e_j \end{cases} \quad (1)$$

In this matrix, each row represents the label of each engine in a scan of a sample, and each column vector C_j represents the decisions made by an engine e_j for all scans. To measure the degree of correlation between two engines, we examine how common the two engines give the same labels to the same samples around the same time. Specifically, given a pair of engines (e_j, e_k), we take their column vectors (C_j, C_k) in matrix R and calculate the Spearman [9] correlation coefficient $\rho_{j,k}$ to measure the correlation. The Spearman correlation coefficient is a non-parametric statistical measure employed to assess the extent to which two variables demonstrate a tendency to increase or decrease concurrently. Coefficients equal to 1 signify a perfect positive correlation, whereas coefficients equal to -1 indicate a perfect negative correlation. Coefficients approaching 0 denote an absence of meaningful correlation between the variables in question. We take a rigorous eye that only if $\rho_{j,k}$ is greater than 0.8 we consider the engines (e_j, e_k) to be strongly correlated.

7.2.1 Overall Level. Taking into account all samples in our dataset, the Spearman correlation coefficients were calculated for each pair of engine label sequences (i.e., any two columns in R). Among these results, we extracted the strong correlations (i.e., the correlation coefficient exceeded 0.8), involving a total of 17 engines. Figure 11 illustrates the network between strongly correlated engines. Specifically, the most significant correlation between the engines *Paloalto* and *APEX* was found, with a coefficient of 0.9933. Moreover, the engines *Webroot* and *CrowdStrike* (0.9754), *Avast* and *AVG* (0.9814), *BitDefender* and *FireEye* (0.9520), *Emsisoft* and *FireEye* (0.9189), *Babable* and *F-Prot* (0.9698), etc., all show strong correlations. This validates that there exist groups of engines whose labeling decisions have strong correlations. Thus, engines' results should not be treated independently. This informs engine selection in many tasks; users can develop more sensible strategies to handle the engine's results for more reliability, e.g., the results from engines with high correlations can be amalgamated and viewed together.

7.2.2 Per File Type Level. The correlation between engines was further examined by considering each file type separately. We find that the correlations between engines vary for different file types. For example, for `DEX` files, there are 6 groups of engines that have strong correlations including *APEX* and *Webroot* (0.8165), *AVG* and *Avast-Mobile* (0.9567), *Ad-Aware* and *Arcabit* (0.8792), *Avira* and *Cynet* (0.9751), *McAfee* and *McAfee-GW-Edition* (0.8301), *Babable* and *F-Prot* (0.9807); while `HTML` files have 49 groups of engines showing strong correlations. Among the different file types, we find that one group of engines (*BitDefener* and *MAX*) show strong correlation with 17 file types, while up to 40 groups of engines show

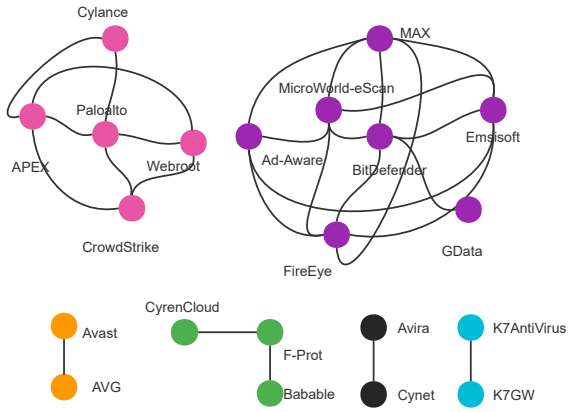


Figure 11: Strong correlations between engines.

strong correlation with only one file type, e.g., *Lionic* and *VirIT* show strong correlation (0.8896) for only GZIP file type. These findings suggest that the correlation between antivirus engines can vary significantly depending on the file type being analyzed. This highlights the importance of considering the file type when analyzing the correlation between antivirus engines. We list the detailed strong correlation between AV engines for some file types in Appendix 2.

Observation 11: *Some antivirus engines have strong correlations in their labeling decisions, impacting engine selection and weighting. Thus engines should not be weighted equally when processing their results.*

8 DISCUSSION

8.1 Implications and Recommendations

Label Stability and Dynamics. We observe that benign samples can stay stable for a longer period of time. This provides justification for VirusTotal detection on benign samples, which are not very susceptible to change. By focusing on fresh dynamic samples, we show that the degree of variation in scan results varies by file type, which enlightens that researchers can apply different criteria when dealing with scan results of different file types. We also show that the degree of variation in the two scan results is related to the time interval between their scans. The two scans with a long time interval are more likely to differ than the two with a short time interval. This is in line with our intuition, and reflects the importance of updating dataset labels in a timely manner. We recommend that researchers should be aware of the inherent variability in AV-Ranks. VirusTotal can consider implementing a notification system for users when significant AV-Rank variations are detected in short time intervals. This can prompt timely re-evaluations.

Label Aggregation. Our re-measurement on label aggregation follows the prior work [36] and we both confirm the validity of the threshold-based aggregation method. However, we have somewhat different insights into the appropriate range of thresholds. Our measurement results are derived from a much larger dataset and are thus more convincing. However, we would like to remark that this finding is a macro-level measure of the overall situation, and may not be perfectly fit for any specific dataset. Researchers might want to consider the ranges we have identified when using the threshold-based aggregation method. However, it is also a good idea to double-check the range's applicability to your specific dataset. Besides, we advocate for continuous updates and refinement of threshold ranges based on ongoing research and user feedback to ensure optimal label stability across various datasets. Collaboration between researchers and platforms like VirusTotal can facilitate this improvement process.

Label Stabilization. Although result changes widely exist in VirusTotal scanning, we show that for most samples, their AV-Ranks and the aggregated labels from the threshold-based method can reach stability after a certain time period. Specifically, over 90% of them reached stability within 30 days. This finding can be useful for relevant researchers and practitioners who rely on VirusTotal scanning for threat detection and analysis, as they can leverage this stabilization period to obtain reliable and consistent results. This also prompts us to suggest that VirusTotal could implement a feature notifying users when a sample's AV-Rank has stabilized, enhancing user trust and result reliability. And this feature could be customizable, allowing users to set their own criteria for what they consider "stable".

Engine Reliability and Correlation. We show that the stability performance of antivirus engines varies significantly across different file types. Even well-known engines like F-Secure and Microsoft showed a significant number of flips, highlighting the need for continuous evaluation and improvement of antivirus engines. Understanding these performance variations across file types is crucial for informed engine selection and optimization for the community. Moreover, we validate that some engines have strong correlations. Users can develop more sensible strategies by amalgamating results from engines that exhibit low correlations to improve accuracy. We recommend that VirusTotal and antivirus providers prioritize continuous evaluation and improvement of engines, especially those with high flip frequencies. As well, consider providing users with correlation data to aid in their engine selection. We also encourage collaboration between antivirus providers to share insights and best practices. This can lead to collective improvements in engine reliability and reduced label flip frequencies.

Comparisons to Prior Work. The present work compares with prior work from many perspectives. Some of the findings align with those of previous research, including the effectiveness of threshold-based label aggregation in tolerating label dynamics, the primary role of engine updates in label changes, and the existence of correlations in engine labeling decisions. However, our research also presents findings that diverge from earlier studies, including recommending different threshold selection ranges and finding that "hazard flips" are surprisingly rare. On top of that, we also delve into areas that were not covered in previous work, such as file type-specific analysis, a detailed examination of stable samples, and an exploration of label result stabilization. Although there are differences between our findings and those of earlier studies, we would like to remark that all of the papers referenced provide excellent contributions and valuable insights. Our objective here is to invigorate this research area and increase community awareness. And we call for more insightful research in this domain, building upon both our findings and those of our esteemed peers.

Measurement Time Window. We can also see from the analysis the importance of having a sufficient measurement time window. For example, without a sufficiently extended time window, it becomes challenging to see the correlation between the AV-Rank difference and the interval between scans, and it becomes difficult to determine whether the labels have really reached stability. On the other hand, we conducted a straightforward assessment to see if the AV-Rank discrepancy obtained for each sample varied across different scan time windows. We observed that when the scan time window for samples from the initial month was extended to three months, 8.6% of samples exhibited a growing AV-Rank gap. And as the scanning window lengthens, the resulting AV-Rank gap distribution of the samples is always variable. Thus, a limited measurement time window may affect some of the analysis results obtained. We recommend an adequate time window for the related measurements.

8.2 Limitations and Threats to Validity

Despite the many novel discoveries, our study carries some potential limitations and threats to validity. First, although a substantial dataset of VirusTotal scan data was collected, we actually used only a small fraction of it for analysis, as the vast majority of samples that had only one scan during our collection period were filtered out. Nevertheless, to the best of our knowledge, this study still involves the largest data size of any existing study. We did not perform any re-scanning because one of our goals is to present an overview of the real scan data in VT. The second threat relates to the measurement of label stabilization. In measuring the stability of the scan results, we consider AV-Rank to be stable if it is constant or fluctuates within a small range, however, there is no standard for how small a range can be considered stable. Third, we recognize that the reproducibility of our study is limited due to the specificity of the dataset used, which entails an inexpensive cost for the VirusTotal premium service. However, this just highlights the value of this dataset, and our general goal is to systematically analyze such a valuable dataset to reveal findings that are not clear enough or not yet known to the community.

9 CONCLUSION

In this paper, we measure and study the dynamics characteristics of VirusTotal scan results using an extraordinarily large-scale dataset. We collect 14 months of VT scan data, comprising a total of 571 million samples and 847 million reports. We addressed some issues related to the dynamics of labels in VirusTotal, including the prevalence of label dynamics/silence, the characteristics across file types, the impact of label dynamics on label aggregation methods, whether scan results become stable over time, and the reliability of individual engines. Our work provides concrete and actionable suggestions for different stakeholders in the community to guide practical improvement efforts.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China (grant No.62072046, 62302181), the Key R&D Program of Hubei Province (2023BAB017, 2023BAB079), and the Knowledge Innovation Program of Wuhan-Basic Research.

REFERENCES

[1] 2022. Statistics - VirusTotal. https://www.virustotal.com/en/statistics/.
[2] 2022. VirusTotal. https://www.virustotal.com/.
[3] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, Konrad Rieck, and CERT Siemens. 2014. Drebin: Effective and explainable detection of android malware in your pocket.. In *Ndss*, Vol. 14. 23–26.
[4] Ulrich Bayer, Paolo Milani Comparetti, Clemens Hlauschek, Christopher Kruegel, and Engin Kirda. 2009. Scalable, behavior-based malware clustering.. In *NDSS*, Vol. 9. 8–11.
[5] Onur Catakoglu, Marco Balduzzi, and Davide Balzarotti. 2016. Automatic extraction of indicators of compromise for web applications. In *Proceedings of the 25th international conference on world wide web*. 333–343.
[6] Kai Chen, Peng Wang, Yeonjoon Lee, XiaoFeng Wang, Nan Zhang, Heqing Huang, Wei Zou, and Peng Liu. 2015. Finding unknown malice in 10 seconds: Mass vetting for new threats at the google-play scale. In *24th {USENIX} security symposium ({USENIX} security 15)*. 659–674.
[7] Euijin Choo, Mohamed Nabeel, Ravindu De Silva, Ting Yu, and Issa Khalil. 2022. A Large Scale Study and Classification of VirusTotal Reports on Phishing and Malware URLs. *arXiv preprint arXiv:2205.13155* (2022).
[8] Fady Copty, Matan Danos, Orit Edelstein, Cindy Eisner, Dov Murik, and Benjamin Zeltser. 2018. Accurate malware detection by extreme abstraction. In *Proceedings of the 34th Annual Computer Security Applications Conference*. 101–111.
[9] Wayne W Daniel et al. 1978. *Applied nonparametric statistics*. Houghton Mifflin.
[10] Yue Duan, Mu Zhang, Abhishek Vasisht Bhaskar, Heng Yin, Xiaorui Pan, Tongxin Li, Xueqiang Wang, and XiaoFeng Wang. 2018. Things You May Not Know About Android (Un) Packers: A Systematic Study based on Whole-System Emulation.. In *NDSS*.
[11] Yu Feng, Saswat Anand, Isil Dillig, and Alex Aiken. 2014. Apposcopy: Semantics-based detection of android malware through static analysis. In *Proceedings of the 22nd ACM SIGSOFT international symposium on foundations of software engineering*. 576–587.
[12] Mariano Graziano, Davide Canali, Leyla Bilge, Andrea Lanzi, and Davide Balzarotti. 2015. Needles in a haystack: Mining information from public dynamic analysis sandboxes for malware intelligence. In *24th {USENIX} Security Symposium ({USENIX} Security 15)*. 1057–1072.
[13] Mahmoud Hammad, Joshua Garcia, and Sam Malek. 2018. A large-scale empirical study on the effects of code obfuscations on Android apps and anti-malware products. In *Proceedings of the 40th international conference on software engineering*. 421–431.
[14] Heqing Huang, Cong Zheng, Junyuan Zeng, Wu Zhou, Sencun Zhu, Peng Liu, Suresh Chari, and Ce Zhang. 2016. Android malware development on public malware scanning platforms: A large-scale data-driven study. In *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 1090–1099.
[15] Alex Kantchelian, Michael Carl Tschantz, Sadia Afroz, Brad Miller, Vaishaal Shankar, Rekha Bachwani, Anthony D Joseph, and J Doug Tygar. 2015. Better malware ground truth: Techniques for weighting anti-virus vendor labels. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*. 45–56.
[16] Doowon Kim, Bum Jun Kwon, and Tudor Dumitraş. 2017. Certified malware: Measuring breaches of trust in the windows code-signing pki. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1435–1448.
[17] Bo Li, Phani Vadrevu, Kyu Hyung Lee, Roberto Perdisci, Jienan Liu, Babak Rahbarinia, Kang Li, and Manos Antonakakis. 2018. JSgraph: Enabling Reconstruction of Web Attacks via Efficient Tracking of Live In-Browser JavaScript Executions.. In *NDSS*.
[18] Vector Guo Li, Matthew Dunn, Paul Pearce, Damon McCoy, Geoffrey M Voelker, Stefan Savage, and Kirill Levchenko. 2019. Reading the tea leaves: A comparative analysis of threat intelligence. In *USENIX Security Symposium*.
[19] Hui Peng, Yan Zhang, Weiping Shi, and Xiaofeng Chen. 2019. Understanding the label process of virus total: A phishing URL perspective. *IEEE Transactions on Information Forensics and Security* 15 (2019), 2221–2234.
[20] Peng Peng, Limin Yang, Linhai Song, and Gang Wang. 2019. Opening the blackbox of virustotal: Analyzing online phishing scan engines. In *Proceedings of the Internet Measurement Conference*. 478–485.
[21] Moheeb Abu Rajab, Lucas Ballard, Noé Lutz, Panayiotis Mavrommatis, and Niels Provos. 2013. CAMP: Content-agnostic malware protection. (2013).
[22] Christian Rossow, Christian J Dietrich, Chris Grier, Christian Kreibich, Vern Paxson, Norbert Pohlmann, Herbert Bos, and Maarten Van Steen. 2012. Prudent practices for designing malware experiments: Status quo and outlook. In *2012 IEEE Symposium on Security and Privacy*. IEEE, 65–79.
[23] Marcos Sebastián, Richard Rivera, Platon Kotzias, and Juan Caballero. 2016. Avclass: A tool for massive malware labeling. In *International symposium on research in attacks, intrusions, and defenses*. Springer, 230–253.
[24] Mahmood Sharif, Jumpei Urakawa, Nicolas Christin, Ayumu Kubota, and Akira Yamada. 2018. Predicting impending exposure to malicious content from user behavior. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 1487–1501.

[25] Linhai Song, Heqing Huang, Wu Zhou, Wenfei Wu, and Yiying Zhang. 2016. Learning from big malwares. In *Proceedings of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems*. 1–8.

[26] Nedim Šrndic and Pavel Laskov. 2013. Detection of malicious pdf files based on hierarchical document structure. In *Proceedings of the 20th Annual Network & Distributed System Security Symposium*. Citeseer, 1–16.

[27] Bo Sun, Akinori Fujino, and Tatsuya Mori. 2016. Poster: Toward automating the generation of malware analysis reports using the sandbox logs. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 1814–1816.

[28] Saravanan Thirumuruganathan, Mohamed Nabeel, Euijin Choo, Issa Khalil, and Ting Yu. 2022. SIRAJ: A Unified Framework for Aggregation of Malicious Entity Detectors. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 507–521.

[29] Ke Tian, Steve TK Jan, Hang Hu, Danfeng Yao, and Gang Wang. 2018. Needle in a haystack: Tracking down elite phishing domains in the wild. In *Proceedings of the Internet Measurement Conference 2018*. 429–442.

[30] Haoyu Wang, Zhe Liu, Jingyue Liang, Narseo Vallina-Rodriguez, Yao Guo, Li Li, Juan Tapiador, Jingcun Cao, and Guoai Xu. 2018. Beyond google play: A large-scale comparative study of chinese android app markets. In *Proceedings of the Internet Measurement Conference 2018*. 293–307.

[31] Liang Wang, Antonio Nappa, Juan Caballero, Thomas Ristenpart, and Aditya Akella. 2014. Whowas: A platform for measuring web deployments on iaas clouds. In *Proceedings of the 2014 Conference on Internet Measurement Conference*. 101–114.

[32] Liu Wang, Haoyu Wang, Xiapu Luo, and Yulei Sui. 2022. MalWhiteout: Reducing Label Errors in Android Malware Detection. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*. 1–13.

[33] Mingyuan Xia, Lu Gong, Yuanhao Lyu, Zhengwei Qi, and Xue Liu. 2015. Effective real-time android application auditing. In *2015 IEEE Symposium on Security and Privacy*. IEEE, 899–914.

[34] Zhaoyan Xu, Antonio Nappa, Robert Baykov, Guangliang Yang, Juan Caballero, and Guofei Gu. 2014. Autoprobe: Towards automatic active malicious server probing using dynamic binary analysis. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. 179–190.

[35] Wei Yang, Deguang Kong, Tao Xie, and Carl A Gunter. 2017. Malware detection in adversarial settings: Exploiting feature evolutions and confusions in android apps. In *Proceedings of the 33rd Annual Computer Security Applications Conference*. 288–302.

[36] Shuofei Zhu, Jianjun Shi, Limin Yang, Boqin Qin, Ziyi Zhang, Linhai Song, and Gang Wang. 2020. Measuring and Modeling the Label Dynamics of Online {Anti-Malware} Engines. In *29th USENIX Security Symposium (USENIX Security 20)*. 2361–2378.

[37] Chaoshun Zuo and Zhiqiang Lin. 2017. Smartgen: Exposing server urls of mobile apps with selective symbolic execution. In *Proceedings of the 26th International Conference on World Wide Web*. 867–876.

Appendix 1 ETHICS

This work does not raise any ethical issues.

Appendix 2 STRONG CORRELATIONS AMONG AV ENGINES

Taking the `Win32 EXE` file type as an example, an undirected graph was constructed, as depicted in Figure 12. we observe that most of the correlations between the engines were consistent with the overall level (see Figure 11). However, there are also some engines such as *Cyren* and *Fortinet* that exhibit strong correlations in `Win32 EXE` files despite not showing a strong correlation at the overall level. Also, *Avira* and *Cynet*, which show strong correlations at the overall level, fail to show strong correlations on the `Win32 EXE` files. Similar analyses can be conducted for other file types to identify the correlations between engines. This comparison of engine performance and relationships across various file types provides valuable insights for engine selection and optimization in diverse contexts.

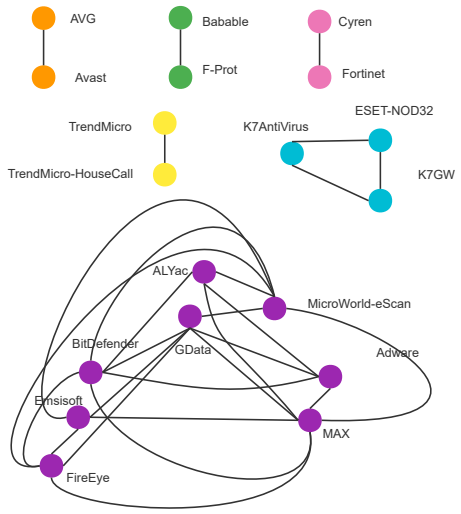


Figure 12: Undirected Connect Graph for Win32 EXE files.

Next, we present the groups of highly correlated engines for the top 5 file types, i.e., Win32 EXE, TXT, HTML, ZIP, and PDF, in Table 4, 5, 6, 7, 8, respectively.

Table 4: Highly correlated engine groups for Win32 EXE

Group	Engines
Group 1	Avast, AVG
Group 2	F-Prot, Babable
Group 3	MicroWorld-eScan, BitDefender, GData, FireEye, MAX, ALYac, Ad-Aware, Emsisoft
Group 4	K7GW, K7AntiVirus, ESET-NOD32
Group 5	TrendMicro-HouseCall, TrendMicro
Group 6	Cyren, Fortinet

Table 5: Highly correlated engine groups for TXT

Group	Engines
Group 1	Avast, AVG
Group 2	K7GW, K7AntiVirus
Group 3	MicroWorld-eScan, BitDefender, GData, FireEye, MAX, ALYac, Ad-Aware, Emsisoft
Group 4	Cynet, Avira
Group 5	F-Prot, Babable
Group 6	Alibaba, Webroot

Table 6: Highly correlated engine groups for HTML

Group	Engines
Group 1	MicroWorld-eScan, BitDefender, GData, FireEye, MAX, ALYac, Cyren, Ad-Aware, Emsisoft
Group 2	Avast, AVG
Group 3	F-Prot, Babable
Group 4	K7GW, K7AntiVirus
Group 5	AhnLab-V3, Cynet, Rising, Cyren, Avira, CAT-QuickHeal, ESET-NOD32, NANO-Antivirus
Group 6	NANO-Antivirus, AhnLab-V3, Rising
Group 7	TrendMicro-HouseCall, TrendMicro
Group 8	Cyren, ESET-NOD32
Group 9	APEX, Webroot

Table 7: Highly correlated engine groups for ZIP

Group	Engines
Group 1	Avast, AVG
Group 2	Cynet, Avira
Group 3	BitDefender, GData, FireEye, MAX, Emsisoft
Group 4	F-Prot, Babable

Table 8: High correlated engine groups for PDF

Group	Engines
Group 1	Avast, AVG
Group 2	MicroWorld-eScan, BitDefender, FireEye, MAX, ALYac, Ad-Aware, Emsisoft
Group 3	F-Prot, Babable
Group 4	Cynet, Avira