Capturing the Behavior of Android Malware with MH-100K: A Novel and Multidimensional Dataset

Hendrio Bragança[1], Vanderson Rocha[1], Lucas Vilanova Barcellos[2], Eduardo Souto[1], Diego Kreutz[2], Eduardo Feitosa[1]

[1]Universidade Federal do Amazonas (UFAM)
[2]Universidade Federal do Pampa (UNIPAMPA)
{hendrio.luis, vanderson, esouto, efeitosa}@icomp.ufam.br
{lucasvilanova.aluno,diegokreutz}@unipampa.edu.br

Abstract. *The fast pace proliferation of Android malware continues to pose challenges to cybersecurity research. To help reshape the future of malware research, we introduce the MH-100K, a dataset that provides a holistic view through 101,975 APK samples, thousands of diverse features and metadata. We use the VirusTotal API to ensure accurate threat evaluation, combining multiple detection methods for precision. Our findings suggest MH-100K is a valuable resource for providing new insights about the malware landscape's evolution.*

1. Introduction

The rapid surge in digital service adoption due to the COVID-19 pandemic has left users vulnerable to an increasing number of malware attacks, resulting in data loss, information theft, and various cybercrimes [Aboaoja et al., 2022, Miranda et al., 2022]. To address the new challenges of this scenario, the role of machine learning (ML) models in malware detection has gained significant traction (e.g., with new advanced research) [Zakeya et al., 2022].

However, the effectiveness of ML models heavily relies on the quality of the dataset used for training. Unfortunately, existing datasets suffer from various limitations, making them outdated and biased for modern malware detection. Recent research has highlighted severe issues such as outdated data, inadequate representation of the studied population, and a limited number of features in commonly used and publicly available datasets [Miranda et al., 2022, Soares et al., 2021b, Soares et al., 2021a].

Such irregularities cast doubt upon the authenticity of reported performances and have the potential to yield misguided conclusions [Miranda et al., 2022]. For instance, the Drebin-215 dataset, released in 2018, is a subset of Drebin, which was developed in 2012 [Soares et al., 2021b]. This means that new models are trained on outdated data that does not accurately reflect the reality of malware from 2018 onward. This reality is substantiated by recent statistics, revealing that lack of adequately representative and high-quality data is among the main culprits behind the failure of up to 80% of AI projects [AI & Data Today, 2023, Schmelzer, 2022].

To address such king of challenges, we introduce the MH-100K dataset[1], which contains 101,975 samples and serves as a foundation for current and future Android malware detection research. Unlike most existing datasets, MH-100K has extensive metadata

[1]https://github.com/Malware-Hunter/MH-100KK-dataset

and comprises a large collection of the most commonly used features to detect Android malware, such as permissions, API calls, and intents. The samples in MH-100K were randomly selected from Androzoo's app list, spanning from 2010 to 2022. We hope this time range allows researchers to achieve a more comprehensive understanding of malware evolution.

We compared the MH-100K dataset with other known datasets in the literature [Bragança et al., 2023], including Drebin-215, Androcrawl, KronoDroid, Android-Permissions, Adroit and DefenseDroid. The wide range of Android malware activities shown by the lack of a direct intersection in feature sets calls for a variety of multidimensional feature sets for comprehensive investigation and detection. The similar behaviors and exploitation techniques of Android malware are nevertheless reflected in the shared categories of features, underscoring the significance of these features in malware identification. Essentially, the features for detecting Android malware are varied and might differ significantly across datasets, reflecting the complexity and dynamic nature of Android malware. This diversity also emphasizes how crucial it is to build a datasets with a wide range of features in order to enable accurate and reliable malware identification.

It is also worth emphasizing that we used the VirusTotal API to label the MH-100K samples, offering a multidimensional view of each sample's threat level. This labeling approach enhances the dataset's richness and provides valuable insights into the performance and reliability of VirusTotal labels.

Our contribution is two-fold. Firstly, we build the MH-100K dataset, which has over 100,000 Android samples and includes a large number of some of the most widely known and used features for malware detection, including permissions, API calls, and intents. Secondly, we provide the first in-depth VirusTotal labeling analysis, discussing the benefits and drawbacks of having a long list of scanners. Our analysis provides a more comprehensive understanding of the robustness of labels obtained using the VirusTotal's metadata.

2. The MH-100K dataset

The MH-100KK dataset is a comprehensive collection of Android malware information, comprising 101,975 samples. It includes a main CSV file with valuable metadata, such as the SHA256 hash (APK's signature), file name, package name, Android's official compilation API, 166 permissions, 24,417 API calls, and 250 intents. Additionally, the MH-100K dataset contains a very rich set of files containing the complete output of the VirusTotal analysis. This information can be used in future research to analyze the patterns of antivirus scan results to understand the prevalence and behavior of different malware families. This analysis can help in creating malware taxonomies, identifying new variants, and studying the evolution of malware over time. We believe MH-100K to be the first comprehensive dataset providing such updated, rich and diverse information to foster state-of-the-art research in advanced Android malware detection.

The samples in the dataset were randomly selected from the extensive list of Android applications available on Androzoo[2]. The sampling period covers a span of 13 years, ranging from 2010 to 2022, providing researchers with the opportunity to study malware evolution and changing characteristics over more than a decade.

[2]https://androzoo.uni.lu/

In short, the creation of the MH-100K dataset was initiated with an early iteration of the ADBuilder tool [Vilanova et al., 2022]. When provided with a list of APK signatures (SHA256 hashes), ADBuilder executes a three-fold procedure encompassing the retrieval of APKs, feature extraction, and VirusTotal analysis. To achieve the richness of the MH-100K, we had to improve the ADBuilder tool. For instance, we stated to collect and store the entire set of metadata provided by VirusTotal online service. Finally, it is also worth emphasizing that we had to use several machines (each one with its own public IP address) and API keys to analyze over 100K samples in approximately two months.

The VirusTotal API[3] is one of the most widely known and used services for detecting suspicious files and URLs. Each request results in a JSON file containing the metadata of one sample. This metadata is required to categorize the sample based on the number of scanners identifying the APK as dangerous. This process ensures an up-to-date and multidimensional view of each sample's threat level, as required by several malware detection methods.

3. Dataset Analysis

We conduct two major evaluations that have a significant impact on the performance of malware detection models. First, we analyze the metadata provided by the VirusTotal service to answer the following question: *How many scanners are required to reliably label a sample and malware?* Currently, VirusTotal has more than sixty scanners for detecting malware in application files, such as APKs. For each APK, VirusTotal will output a list of scanners that positively identify it as malware. Finding out the number of scanners threshold is crucial for the second stage of our investigation, in which we measure the impact of various feature numbers on the performance of our machine learning model. We use the XGboost classifier (Extreme Gradient Boosting), which has been used in several works related to malware detection [Palša et al., 2022].

3.1. Labelling threshold

We used 1 to 8 scanners in our experiment to label the samples in our dataset. For instance, we consider an APK as malware only if it was detected as such by 8 or more scanners while creating the dataset with labels assuming a minimum of 8 scanners. We use XGBoost as a ranking algorithm to find out the optimal number of scanners to evaluate label quality.

We summarize our findings in Figure 1, showing the confusion matrix for each subset varying the number of scanners from 1 (1-class) to 8 (8-class). The $benign$ class's precision generally increases with the number of scanners, ranging from 0.94 for one scanner to 0.97 for eight scanners. According to this pattern, increasing the number of scanners improves the model's ability to classify benign applications while decreasing false positives correctly. With the exception of one scanner, the recall for the $benign$ class remains consistently strong, suggesting that most authentic applications are correctly identified. These patterns are reflected in F1-score, which combines precision and recall.

In contrast, the performance metrics for the $malware$ class reveal more uncertainty as the number of scanners increases. Precision shows rapid improvement, achieving a high of 1.00 for six scanners. During the same time frame, however, the recall drops

[3]https://developers.virustotal.com/reference/overview

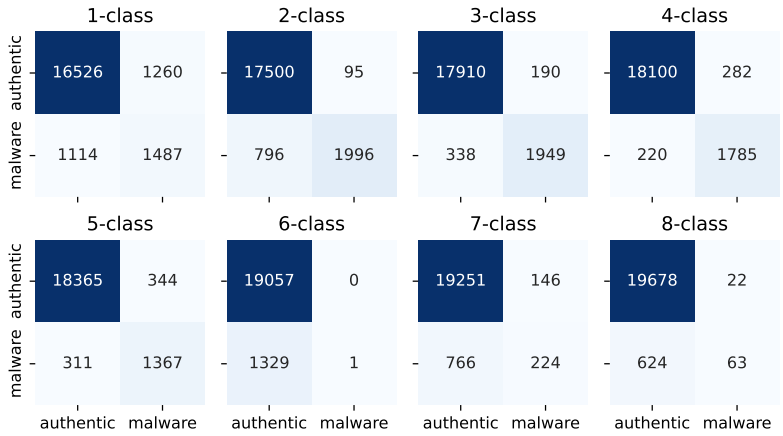

Figure 1. Confusion matrices for labeling samples using 1 to 8 scanner detection.

dramatically to 0.00, indicating that, while the model did not misclassify benign applications as malware, it could not reliably detect any actual malware instances. This results in an F1-score of 0.00, showing that despite its perfect precision, the model's performance is poor due to its failure to identify any malware correctly.

Based on our findings, we recommend using 4 scanners as a good number for this model, with both classes having slightly higher precision, recall, and F1-scores. However, despite the lower recall for the $malware$ class, using more scanners may be advantageous if lowering false positives is a top concern.

3.2. Model Performance vs Number of Features

In our second experiment, we set the labelling threshold at 4 scanners and investigate the impact of varying the number of features (from 64 to 12000) on the XGboost classifier. Our results for precision, recall, f1-score, accuracy and macro-f1 are in Table 1.

As a first observation, both $benign$ and $malware$ classes continuously exhibit high precision, recall, and f1-score across the entire feature size range, indicating the model's ability to effectively classify both malicious and benign applications. The precision, recall, and f1-score metrics are all very similar across all feature sizes, especially when focusing on the $benign$ class. This consistency shows that, despite the number of features included, the model consistently performs well at correctly identifying benign applications.

On the other hand, our results show a slightly different pattern for the $malware$ classes. As the number of features increases, most significantly from 64 to 1000, there is an increase in both recall and f1-score but precision remains essentially the same. The model's capacity to correctly identify malware instances has improved with the increase in feature size.

The models' accuracy remains consistent at 97-98% across all feature sizes. This demonstrates that, regardless of feature size, the models are generally successful in accurately categorizing both malware and benign instances. The Macro-f1 score, an important metric in scenarios with an imbalanced class distribution, improves from 0.91 to 0.94, corresponding with the increase in feature size. This improvement implies that the model's

Table 1. Performance metrics varying the number of features.

Feature	Class	Precision	Recall	F1-score	Accuracy	Macro-F1
64	authentic	0.98	0.99	0.98	0.97	0.91
	malware	0.88	0.81	0.84		
128	authentic	0.98	0.99	0.98	0.97	0.92
	malware	0.88	0.82	0.85		
256	authentic	0.98	0.99	0.98	0.97	0.92
	malware	0.87	0.84	0.85		
512	authentic	0.98	0.99	0.98	0.97	0.92
	malware	0.87	0.84	0.86		
1000	authentic	0.99	0.99	0.99	0.97	0.93
	malware	0.87	0.87	0.87		
2000	authentic	0.99	0.98	0.99	0.98	0.93
	malware	0.86	0.89	0.88		
4000	authentic	0.99	0.99	0.99	0.98	0.94
	malware	0.87	0.90	0.89		
6000	authentic	0.99	0.99	0.99	0.98	0.94
	malware	0.87	0.91	0.89		
8000	authentic	0.99	0.99	0.99	0.98	0.94
	malware	0.87	0.91	0.89		
10000	authentic	0.99	0.99	0.99	0.98	0.94
	malware	0.87	0.90	0.88		
12000	authentic	0.99	0.99	0.99	0.98	0.94
	malware	0.87	0.91	0.89		

balanced performance across both classes will improve as feature size increases.

The conclusions drawn from these findings demonstrate how well our ML model works at identifying Android malware in the MH-100K dataset. Larger feature sizes, although more effective, may come at a cost in terms of increased computing complexity and expense. Regardless, the improved speed of the $malware$ class with larger feature sizes is an essential consideration in scenarios when detecting malicious instances is critical.

4. Future Directions

We introduced the MH-100K dataset, a large collection of more than 100,000 Android samples, making a substantial addition to the field of Android malware detection. Our dataset offers a comprehensive resource for building reliable machine learning models and contains a variety of frequently used features for malware detection, such as permissions, API calls, and intents.

Our findings support the capability of our ML model to recognize Android malware using the MH-100K dataset. Although performance can be improved by employing larger feature sizes, it should be noticed that this may result in higher computational complexity and cost. However, the noticeable increase in malware detection rate with higher feature sizes is an important factor to take into consideration, particularly in situations where rapid detection of potentially dangerous events is critical. Further research is encouraged to investigate the ways to control the increased computational complexity and

cost driven by higher feature sizes, potentially by using more effective feature selection or extraction methods.

Acknowledgement

This research was funded, as provided for in Arts. 21 and 22 of decree no. 10,521/2020, under Federal Law no. 8,387/1991, through agreement no. 003/2021, signed between ICOMP/UFAM, Flextronics da Amazônia Ltda and Motorola Mobility Comércio de Produtos Eletrônicos Ltda. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES-PROEX) - Finance Code 001 and by Amazonas State Research Support Foundation - FAPEAM - through the POS-GRAD project.

References

Aboaoja, F. A., Zainal, A., Ghaleb, F. A., Al-rimy, B. A. S., Eisa, T. A. E., and Elnour, A. A. H. (2022). Malware detection issues, challenges, and future directions: A survey. *Applied Sciences*, 12(17):8482.

AI & Data Today (2023). Top 10 reasons why ai projects fail. https://www.aidatatoday.com/top-10-reasons-why-ai-projects-fail.

Bragança, H., Rocha, V., Souto, E., Kreutz, D., and Feitosa, E. (2023). Explaining the effectiveness of machine learning in malware detection: Insights from explainable AI. In *Anais do XXIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, Porto Alegre, RS, Brasil. SBC.

Miranda, T. C., Gimenez, P.-F., Lalande, J.-F., Tong, V. V. T., and Wilke, P. (2022). Debiasing android malware datasets: How can i trust your results if your dataset is biased? *IEEE Transactions on Information Forensics and Security*, 17:2182–2197.

Palša, J., Ádám, N., Hurtuk, J., Chovancová, E., Madoš, B., Chovanec, M., and Kocan, S. (2022). Mlmd—a malware-detecting antivirus tool based on the xgboost machine learning algorithm. *Applied Sciences*, 12(13):6672.

Schmelzer, R. (2022). The one practice that is separating the ai successes from the failures. Forbes. https://www.forbes.com/sites/cognitiveworld/2022/08/14/the-one-practice-that-is-separating-the-ai-successes-from-the-failures/?sh=33b3f6a17cb5.

Soares, T., Mello, J., Barcellos, L., Sayyed, R., Siqueira, G., Casola, K., Costa, E., Gustavo, N., Feitosa, E., and Kreutz, D. (2021a). Detecção de Malwares Android: Levantamento empírico da disponibilidade e da atualização das fontes de dados. In *VI WRSeg*.

Soares, T., Siqueira, G., Barcellos, L., Sayyed, R., Vargas, L., Rodrigues, G., Assolin, J., Pontes, J., Feitosa, E., and Kreutz, D. (2021b). Detecção de Malwares Android: datasets e reprodutibilidade. In *VI WRSeg*.

Vilanova, L., Kreutz, D., Assolin, J., Quincozes, V., Miers, C., Mansilha, R., and Feitosa, E. (2022). ADBuilder: uma ferramenta de construçao de datasets para detecçao de malwares android. In *Anais Estendidos do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 143–150. SBC.

Zakeya, N., Ségla, K., Chamseddine, T., and Alvine, B. B. (2022). Probing androvul dataset for studies on android malware classification. *Journal of King Saud University-Computer and Information Sciences*, 34(9):6883–6894.