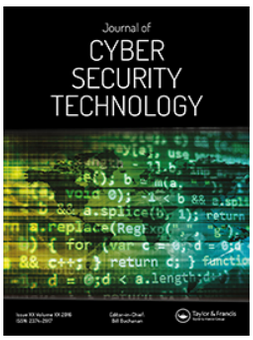


Journal of Cyber Security Technology

ISSN: 2374-2917 (Print) 2374-2925 (Online) Journal homepage: http://www.tandfonline.com/loi/tsec20

Empirical study of malware diversity in major Android markets

César Soto-Valero & Mabel González

To cite this article: César Soto-Valero & Mabel González (2018): Empirical study of malware diversity in major Android markets, Journal of Cyber Security Technology

To link to this article: https://doi.org/10.1080/23742917.2018.1483876

Published online: 27 Jun 2018.

JOURNAL OF CYBER SECURITY TECHNOLOGY
https://doi.org/10.1080/23742917.2018.1483876

Taylor & Francis
Taylor & Francis Group

Empirical study of malware diversity in major Android markets

César Soto-Valero and Mabel González

Department of Computer Science, Universidad Central "Marta Abreu" de Las Villas, Santa Clara, Cuba

ABSTRACT

The popularity of Android has motivated a significant increase in the amount of malware specially designed to target this operating system. During the last years, the threat has become more serious and every day cybercriminals create and share new specimens through almost all existing markets. This situation has promoted a notable research interest in the development of automated malware detection and classification systems. In this paper, we perform a large-scale empirical study to examine the diversity of Android malware in major markets. Through the analysis of more than 5 million of apps, we use the labels assigned by 57 different anti-malware vendors and diversity measures to get insights about the distribution and evolution of Android malware. Furthermore, we propose a dissimilarity measure for comparing these labels, which can be applied as part of an agglomerative hierarchical clustering algorithm. This clustering method groups the labels according to the scanning reports of different anti-malware vendors. The results obtained make evident an increase in the diversification of malware in both official and alternative markets. Moreover, we show how the criteria of various anti-malware, in conjunction with clustering techniques, is a suitable approach for grouping and analysing malware samples that perform a similar behaviour.

ARTICLE HISTORY
Received 28 November 2017
Accepted 19 May 2018

KEYWORDS
Android OS; malware analysis; diversity measures; anti-malware scanning results; agglomerative hierarchical clustering

1. Introduction

In the last years, we have witnessed an increasing spread in the use of smartphones and, as consequence, the emergence of a large number of virtual markets for the distribution of mobile applications ('apps', for brevity). At present, Android is the world's most widely used operating system for mobile devices [1]. The growing popularity of Android arises from two core features: its open source nature and its well-designed software architecture. These advantages leveraged a rise in the development of new apps for satisfying the most specifics users' needs. Today, more and more users that rely on Android devices are able to install third-party apps from official and alternative virtual markets. Thus, the security of their devices and the underlying distribution networks become an essential concern for both the end users and their service providers.

The increasing complexity of the Android operating system, in conjunction with the sensitive data stored in mobile devices, makes the platform an attractive attack surface and lucrative target for cybercriminals [2]. This issue has not gone unforeseen by malware

CONTACT César Soto-Valero cesarsotovalero@gmail.com

authors, and the growth of malware targeting Android has increased steadily in the last few years [3].

As a consequence, practitioners and researchers have observed the emergence of a variety of new Android malware families (e.g. GRAYWARE [4] and PACKAGED [5]). The many existing threats range from simple unwanted advertisement, user tracking and disclosure of personal information, to advanced fraud and premium-rate SMS services subscription, or even unwarranted involvement in botnets attacks.

The high percentage of malware targeting the Android system is partly due to the lack of security checks performed when the apps are being distributed [6]. Although most users are nowadays aware that personal computers can be attacked by dangerous viruses, few people realize that their smartphone is prone to a similar threat. Therefore, due to the large number of applications available in the Android markets, there is a natural need of methods that allow to search and detect malicious apps.

The continuous rise of threats targeting Android have attracted the attention of the software security research community [3,7–9]. Previous research on malware in popular Android markets indicated that the majority of malicious apps are distributed through alternative marketplaces [3,6]. Current malware authors are using advanced techniques such as malicious code integration and obfuscation to avoid further detection [10]. The penetration and prevalence of Android malware in major markets, such as Google Play, can compromise a large number of smartphones and cause several damages to the users.

To face the pervasive threat of malicious software, generally downloaded from the internet, prudent users rely on the scan results yielded by anti-malware systems. Unfortunately, each anti-malware vendor has its own secret recipe on how or why it decides to flag as malware a suspicious software application. Therefore, a malware can be detected and classified in a different way by several anti-malware products, causing labelling confusions [11]. Nevertheless, both professionals and researchers rely on anti-malware decisions, whether to flag suspicious apps or to build ground truths for performing malware analysis tasks.

In this context, the diversity of malware, as a reflection of its evolution, is a major concern of practitioners, scientists and researchers working in the field of cyber security of mobile devices. In this paper, we perform a large-scale empirical study on the natural diversity of Android malware in major existing virtual markets. We analyse a dataset with more than 5 million of Android apps, provided by the AndroZoo[1] research project, using various diversity measures. These measures are based on the scanning results of 57 anti-malware vendors hosted by the VirusTotal[2] web service, which is an online platform that can check uploaded files against several commercial and renown anti-malware engines (i.e. Symantec, Avast, Kasperky, McAfee, Panda and others).

This study goes beyond the diversity analysis of apps flagged as malware. We investigate for details about the relation between types and family names of Android malware, as well as the labelling results of anti-malware products according to their scanning reports. We hypothesize an increasing trend in the diversity of Android malware in major markets, as well as the inefficacy of several anti-malware products to detect dangerous apps. Summarizing, this paper makes the following contributions:

- We present an extensive empirical study of malware diversity in major Android markets and describe the diversification and evolution of malware from various perspectives.

- We propose the application of a new dissimilarity measure, based on the notion of multisets, as a useful approach for comparing the malware labelling results of various anti-malware engines.
- We perform hierarchical clustering in order to analyse the relation among the labels assigned by different anti-malware vendors, grouping similar malware samples according to their scanning results.

There is scarce research work on the diversity of malware using the reports of anti-malware scanning. Our study aims to fill this gap by conducting such analysis and reporting our findings based on a large dataset. This work contributes to get insights about the evolution and growing of malware. It also shows the necessity of using multi-criteria-based detection systems, in order to deal with this dangerous type of diversity. To our knowledge, this is the first attempt of studying the presence of malware in Android markets using diversity measures.

The rest of this paper is structured as follows. Section 2 offers a review on software diversity and malware analysis. Section 3.1 provides detailed information about the data sets utilized in this study, as well as the tool used for unifying the malware labels. Sections 3.2 and 3.3 present the diversity measures applied and the dissimilarity function proposed, respectively. Section 4 depicts the main findings of our experimental study, while Section 5 offers a discussion on such results. Section 6 presents the related work, and Section 7 summarizes our findings and concludes the paper.

2. Background

2.1. *Software diversity*

In the last decade, an increasing number of research projects on software engineering have explored the effects of diversity in complex computational systems. These approaches are based on existing diversity concepts from biological sciences, but also have their own particular characteristics. The phenomenon of software diversification can be viewed from various perspectives (e.g. as an obfuscation technique, or via for increasing the robustness and adaptability of software systems). In general, the main goal of software diversity is to promote adaptive capacities in the face of unforeseen structural and environmental variations [12].

Malware is software, too. As occurs in general (goodware) software, malware diversity persists under many different conditions. The variety of malware makes harder for anti-malware systems to detect the presence of new threats. From the perspective of the malware, it is convenient to not manifest an obvious behaviour and to be constantly changing itself. The sophistication and diversification of malware has promoted the emergence of metamorphic malicious engines, where the malware shows different forms according to the dynamic environment [13].

Several forms of software diversity have been identified and properly defined in the literature [14]. In this paper, we are interested in assessing natural diversity, which is a form of software diversity that spontaneously emerges from its continuous development [15]. Specifically, our study is focused on analysing the natural diversity of Android malware.

We explore four dimensions of diversity known as variety, ubiquity, balance and disparity [16]. Our purpose is to use these diversity measures to quantify the impact and prevalence of malware in major Android markets. We are interested in analysing the

impact of such diversity regarding the different types and families of malware. The analysis of malware diversity is important in the field of cyber security research because contributes to the development of better malware analysis and detection systems by studying the evolution of specific variants during time.

2.2. *Automatic malware labelling*

In the context of malware detection and analysis, it is a useful practice to group the samples according to their observed behaviour. This allows researchers and practitioners in cyber security technologies to focus on analysing more specific types of threats [6,17]. Table 1 shows illustrative examples, including a brief description, of some of the most common types of Android malware.

A practical way to conduct malware analysis consists in assign a set of predefined labels to groups of samples according to their observed behaviour. In addition to the binary detection result regarding the maliciousness or not of a scanned sample, anti-malware engines also provide, in case of positive detection, a string report which indicates the type, family and perceived behaviour of the malware under analysis. It is expected that such scanning reports specify the threat appropriately, in a meaningful and consistent way [11]. However, each anti-malware has its own list of labels and there is no universal consensus about the structure of the labelling reports.

Recently, we have witnessed the growing interest of researchers and practitioners in the development of automatic unifying systems to label Android malware using various anti-malware reports (e.g. AVClass [18] and Euphony[19]). This has facilitated the classification and analysis of new threats at large scale.

In general, a malware label contains four main features: (1) platform (the operating system for which the threat is designed, i.e. WINDOWS, ANDROID, etc.), (2) type (the kind of threat, i.e. TROJAN, WORM, etc.), (3) family (the group of threats to which it is associated in terms of behaviour, i.e. GRAY, DROIDKUNGFU, etc.) and (4) extra information (a description of the threat, including its variants). An example of labelled malware, where TROJAN represents its type and DROIDKUNGFU represents its family name, is the following:

< ANDROID>< TROJAN>< DROIDKUNGFU>*<"Collects a variety of information on the infected phone and dump it to a local file which is sent to a remote server afterwards">*

The automatic labelling of malware according to its behaviour offers several benefits. For example, it facilitates the creation of generalized signatures for mitigation tasks, identifying

Table 1. Description and examples of the most common threats exhibited by Android malware.

Type	Description	Examples
ADWARE	Sends personalized advertisements based on user's collected data such as location, web browsing or media. It is mainly focused in the obtaining economical revenues.	SHEDUN, FAKEAPP. AL, MOBIDASH
TROJAN	Masquerades as a benign app to hide its maliciousness identity. It feigns useful functionalities to the user but performs malicious activities in the background without user's consent.	FAKEBANK, FONCY, ANSERVERBOT
BACKDOOR	Enables remote access to the system by bypassing its authentication mechanisms. It usually exploits vulnerabilities in the system to take root's privileges, having the ability to hide itself and remain undetected.	XSIDER, LUCKYCAT, PJAPPS
SPYWARE	Sends personal user's data such as contact list, messages, location, moves and other confidential data to a remote server.	SMACK, BIIGE, PLANKTON
RANSOMWARE	Encrypts files or locks the system to make it inaccessible, and only decrypt or unlock it until some ransom is paid by the victim.	FAKEDEFENDER, LOCKER, SVPENG

if a malware sample found is an instance of a well-known family or a sample of a new malware variant, selecting disinfection mechanisms, attribution, malware lineage, etc.

Such a labelling process can be accomplished manually by experts in cyber security or automatically using clustering techniques based on Machine Learning (ML) [20,21]. While the assignment of malware labels is greatly facilitated by online services such as VirusTotal, grouping malware into families is not an easy task. The main reason for this difficulty is the lack of a standard naming contract in the industry (conventions such as CARO[3] are not widely used). In the absence of common standards, researchers must deal with a plethora of different naming schemes and the lack of labelling consensus among anti-malware vendors, which makes difficult the collection of malware samples in the form of consistent datasets for its empirical research.

2.3. *Malware clustering*

Clustering is an unsupervised ML technique that has been used in exploratory data analysis for identifying groups (clusters) of instances that exhibit similar characteristics. The similarity between instances is commonly defined using some specific criteria (e.g. inter-observation distance measures or correlation-based distance measures). In a nutshell, the goal of clustering methods is to maximize the similarity of the intracluster (internal homogeneity) and minimize the similarity of the intercluster (external separation).

Over the last years, many works have explored the possibility of grouping malware samples into classes according to some structural behavioural-based similarity measures [22]. In the software security context, clustering methods have been also widely used for identifying shared attributes among different variants of malware [20,23,24]. These attributes can be used later to automatically characterize new specimens, thus facilitating their subsequent analysis.

Another interesting application of clustering methods consist in grouping malware samples according to the scanning results of various anti-malware engines, each with its own heuristic detection system. Indeed, nowadays anti-malware vendors remain the most trusted approach to flag an app as malware and associate a label to each malware. In this work, we describe an agglomerative hierarchical clustering approach to group the labels assigned to Android malware using the scanning reports of various anti-malware vendors.

Agglomerative hierarchical clustering (a.k.a. AGNES) is a method for clustering similar elements according to a similarity measure that must be defined previously [25]. AGNES works in a bottom-up manner, this is, each object is initially considered as a single cluster element (leaf). At each step of the algorithm, the two more similar clusters are combined to form a new bigger cluster (node). The procedure is iterated until all points are member of just one single big cluster (root). The result is an informative structure in the shape of a tree, called dendrogram, which shows how the clusters evolved.

3. Materials and methods

3.1. *Data description*

Android apps are currently being distributed through the official Google Play market and also through many of the so-called alternative markets. This study is based on a large

dataset that comprises a total of 5,295,482 Android apps provided by the AndroZoo research project. In addition, we enrich our dataset with the reports provided by a total of 57 anti-malware engines hosted in the VirusTotal web service. The final malware labels were inferred using the Euphony unification tool. In this section, we offer more details regarding the preprocessing and data manipulation.

3.1.1. *Androzoo dataset*

The AndroZoo data set represents a valuable research effort that consists in the collection of millions of Android apps from various data sources [26]. Through the AndroZoo HTTP API, it is possible to download a complete and unaltered collection of apps for analysis. The dataset is freely available to the software security and research community, which contributes to perform more generalizable, reliable and reproducible research studies based on a broad set of representative and updated collection of Android apps.

These apps were obtained by crawling well-known app stores, including the official Google Play market. In addition, all the collected apps were analysed by 57 different anti-malware products using the VirusTotal web service. Table 2 shows the information fields available in the AndroZoo dataset.

The AndroZoo dataset has already been used to conduct researches in the field of automatic malware detection. In practical terms, the size of this dataset allows to demonstrate methodological issues when evaluating the performance of ML detection methods [27]. For example, to assess the time performance of new malware detection algorithms [28] or to obtain a more general landscape of the Android malware from a forensic perspective [2]. It also allows to track the evolution of Android apps over time to find common design patterns [29] or detecting privacy leaks [30]. Other potential uses of the AndroZoo dataset are the code recommendation, to perform large-scale studies on APIs usage, coding patterns, repackaging detection, library adoption, obfuscation techniques, analysis of similarities between apps, etc.

3.1.2. *Euphony labelling tool*

Euphony is a command-line tool for the unification of malware labels of Android; details of the tool are presented and discussed by Hurier et al. [19]. The main purpose of Euphony is to infer a single malware type and family name (or just 'family', for short) per malicious app. The

Table 2. Description of AndroZoo data fields for one example of app from the Google Play market.

Field name	Description	Data type	Example
sha256	The unique hash code that identifies the app.	String	000135C67F6FAE5F1.
dex_date	The app compilation date as provided in the *.dex* file.	ISO Date (UTC)	2016–03–03 17:57:06
apk_size	The size of the app in bytes.	Integer	32,861,192
pkg_name	The name of the main package of the app.	String	kvp.jjy. MispAndroid320
vercode	The version code of the app.	Integer	144
vt_detection	The number of anti-malware vendors from VirusTotal that flag it as malicious.	Integer	3
vt_scan_date	The date in which the app was submitted to VirusTotal.	ISO Date (UTC)	2016–04–06 09:30:59
dex_size	The size of the *.dex* file.	Integer	6,316,108
market	The market where the app was downloaded from.	String	play.google.com

inference can be derived from the results of VirusTotal reports. For each app sent to VirusTotal, the tool returns two relevant pieces of information as follows:

- A binary flag (True = positive detection, False = negative detection)
- A string label that identifies the threat (e.g. 'TROJAN:ANDROIDOS/GINGER MASTER.A')

The Euphony tool can handle the inconsistencies observed in anti-malware labels, allowing a fine-tune of the string differences that refers to the same families. It attempts to break the gap that requires practitioners to have a well-described reference of malware to handle new and unknown samples.

Figure 1 illustrates the three main labelling steps performed by Euphony. We additionally show the content of two important resultant files, which are obtained from stages 2 and 3 of the process. First, the tool takes as input a set of anti-malware scanning reports from VirusTotal. Next, the label fields are extracted from these reports based on the type and family name that was assigned by each individual anti-malware to the sample. At this step, a *.json* file is created with the mapping information of the assigned labels and it frequencies per anti-malware. Then, during the stage 3, Euphony infers the relations among the election labels, bringing a general consensus among the different vendors and returning the most appropriate type and family to the given sample.

Euphony allows to gather useful information about Android malware, including the syntactic and semantic associations between similar malware labels (e.g. basebridge, basebrid, etc.). This could be useful, for example, to create a single target class prior to the experimentation with supervised ML methods for malware detection. Euphony also offers an interface with the public HTTP API of the AndroZoo project through the 'sha256' field (see Table 2). This allows the creation and update of reference datasets of labelled malware, which is important to the research community for the realization of more reliable and reproducible experiments.

In this study, we build our dataset by joining the Euphony's *.json* resultant files, shown in Figure 1, with the AndroZoo. This provides us a vast source of information for the development of our diversity analysis of Android malware. We collected the labels of 5,295,482 Android apps, exploring the data using diversity measures and characterize the malware in major Android markets.

3.2. *Diversity measures*

Some diversity measures combine several aspects of diversity by varying their parameters. We follow the definitions used by Guevara et al. [31], associating the terms entity and category to an Android market and a class of malware, respectively. The purpose of entities consists in describing the systems or agents that host a set of categories (malware types or families, in this case). In the following, we describe in more detail the diversity measures used in this study, which were adapted for quantifying the diversity in our dataset of Android malware.

3.2.1. *Variety and ubiquity*

The variety or richness is commonly the first approach to measure diversity in complex systems. It consists in counting how many categories or types an entity has. Closely related to variety is the concept of ubiquity or rareness, which represents the variety of entities that each category has.

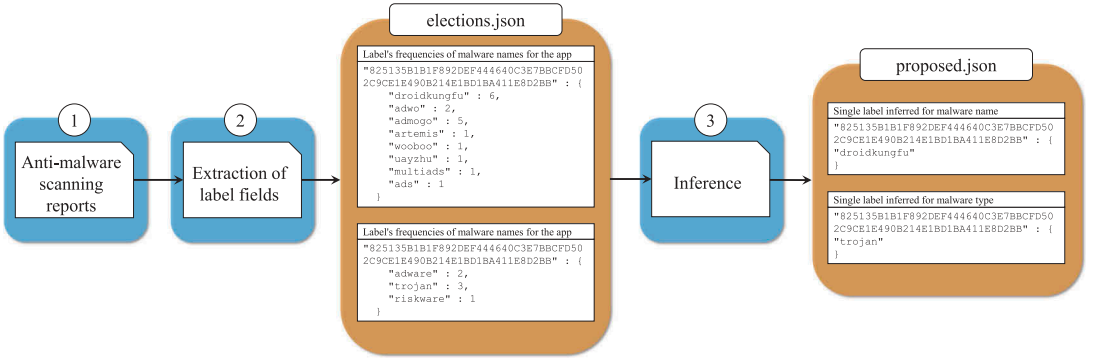


Figure 1. Representation of the main steps for inferring malware labels using the Euphony tool. The *elections.json* and *proposed.json* files offer information regarding the labels given by anti-malware engines and the inference proposed by Euphony.

We use the concepts of variety (V) and ubiquity (U) to measure the distributions of types and families of Android malware in the studied markets. Equations 1 and 2 show how these measures are calculated according to our particular research interests:

$$V_m = \sum_{i=1}^{n} p_i, \tag{1}$$

where n represents the number of different classes of malware and $p_i \in \{0, 1\}$ implies the absence or not of the malware i in the market m.

$$U_i = \sum_{m=1}^{r} q_m, \tag{2}$$

where r represents the number of studied markets and $q_m \in \{0, 1\}$ implies the absence or not of the malware i in the market m.

3.2.2. *Balance*

Balance is a diversity measure used for quantifying how much of each category an entity has. Measuring balance alone is not an easy task due to the difficulty to remove the effect of variety on it [32]. Diversity measures related to the balance property could be associated to the statistical concept of dispersion.

The Shannon Entropy (*SE*) is a traditional measure of dispersion in Information Theory [33], which has been adopted as a standard diversity measure for quantifying balance in complex systems. In this work, we use *SE* to calculate the balance of malware in Android during a given time period. A high value of *SE* implies more diversity in the variants of malware, while a low value of *SE* means more concentration of similar threats. Equation 3 shows how *SE* is calculated for a specific time period:

$$\mathrm{SE}_t = -\sum_{i=1}^{n} p_i \log_2 p_i, \tag{3}$$

where p_i is the ratio between the number of malware i and the total malware that exist in the major Android markets at the time t.

3.2.3. *Disparity*

Another important dimension of diversity is the disparity or dissimilarity between categories or entities [16]. Disparity provides a notion of how different are the categories of a given entity. This concept is closely related to distance metrics and similarity measures such as the Euclidean Distance, Cosine Similarity or Jaccard Index.

The Rao-Stirling (*RS*) is a flexible measure that allows quantifying disparity based on a dissimilarity function [34]. It consists in the sum of the multiplication of the distances (disparity) and the proportions (balance) between two pairs of distinct categories i and j. The *RS* measure uses the parameters α and β for weighting the level of importance assigned to the disparity or balance, respectively. Equation 4 shows how *RS* is calculated for performing our analysis of malware diversity:

$$RS_{\mathrm{m}} = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{i,j}{}^{\alpha} (p_i p_j)^{\beta} \tag{4}$$

where p_i and p_j are the number of variants of malware i and j present in the market m, $d_{i,j}$ is the dissimilarity matrix obtained (e.g. using Cosine distance) between i and j, and n is the total number of markets; $0 < \alpha, \beta < 1$.

3.3. *Dissimilarity measure*

We need to define a dissimilarity measure in order to compare malware samples for subsequent clustering analysis. We use the malware labels assigned by a set of anti-malware engines to the Android apps. Since each app could be flagged by more than one anti-malware and labelled in more than one way, we propose the use of a dissimilarity measure based on the mathematical notion of multisets.

Let $P = (p_1, ..., p_i, ..., p_n)$ be a set of n apps. Let $A = (a_1, ..., a_j, ..., a_m)$ be a set of m anti-malware engines. Then, the labelling result of each anti-malware vendor, for a given app, could be represented as an $n \times m$ matrix as is shown in Table 3, where $l_{i,j}$ corresponds to the string label assigned to the app p_i by the anti-malware a_j.

If the app p_i is not flagged positively as a threat by the anti-malware a_j, then the element $l_{i,j} = \emptyset$. Let $(l_{i,1}, l_{i,2}, ..., l_{i,m})$ be the i^{th} row vector of the matrix, corresponding to the app p_i, then the matrix can be vectorized as $\Psi = (s_1, ..., s_i, ..., s_n)$, where s_i is the set of labels assigned to the app p_i (excluding null values). The vectorized representation allows to retrieve all labels assigned to a given app p_i by a set of m anti-malware vendors.

A multiset is defined as a set of elements with repeated values [35]. More formally, given a set C, a multiset consists in a mapping $m : C \to \mathbb{Z}_{\geq}$, where $m(c, C)$ represents the multiplicity of the element $c \in C$. Multiset functions have identical counterparts as usual functions on sets, but the multiset functions take into account the multiplicity of its elements [36]. Thus, if X and Y are both multisets with $x \in X$ and $y \in Y$, $i > 0$, then the following operations can be defined on them:

- The union $X \cup Y$ is the multiset whose unique elements are the unique elements of X and Y, where the multiplicity of elements in the result is $max(m(x, X), m(y, Y))$.
- The intersection $X \cap Y$ is the multiset similar to union, but the multiplicity of elements in the result is $min(m(x, X), m(y, Y))$.
- The difference $X - Y$ of multisets is similar to union and intersection, in this case the multiplicity is calculated as $max(0, m(x, X) - m(x, Y))$, that is, the element x is in the result if it has more occurrences in X than in Y, and it occurs with the multiplicity of the difference of the left and right multiplicities.

Table 3. Representation of the labels assigned by a set of m anti-malware vendors to n apps.

	a_1	a_2	...	a_m
p_1	$l_{1,1}$	$l_{1,2}$	...	$l_{1,m}$
p_2	$l_{2,1}$	$l_{2,2}$	...	$l_{2,m}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
p_n	$l_{n,1}$	$l_{n,2}$	...	$l_{n,m}$

For instance, the set of label types assigned to the app of Figure 1 could be represented in the form of a multiset as (ADWARE, ADWARE, TROJAN, TROJAN, TROJAN, RISKWARE), which can be abbreviated to ($ADWARE^2$, $TROJAN^3$, $RISKWARE^1$) and by leaving out the element names, we obtain the vector of multiplicities $(2, 3, 1)$ in a three-dimensional space.

Let $X, Y \in \Psi$ be two vectors of labels, then we can manipulate both elements as multisets based on the operations described above. Equation 5 shows our adaptation of the Jaccard distance function for measuring the dissimilarity between multisets X and Y of labelled apps:

$$mdist(\mathrm{X}, \mathrm{Y}) = 1 - \frac{\sum_{i=1}^{n} min\{|X \cup Y| - |X \cap Y|\}}{\sum_{i=1}^{n} max\{|X \cup Y|\}} \tag{5}$$

where $X \cap Y$ represent the intersection of the multisets of labels X and Y, $X \cup Y$ is the union, and min/max retrieve its minimum and maximum multiplicity, respectively.

4. Results

In this section, we present our findings regarding the diversity of Android malware in major markets. We first perform a diversity analysis of malware types and families in the data set described in Section 3.1 using the diversity measures presented in Section 3.2. Next, we use the labels given by anti-malware engines in order to cluster similar malware samples according to the dissimilarity measure proposed in Section 3.3.

4.1. *Diversity analysis*

Figure 2 shows the 10 more representative markets in the data set in terms of the number of registered apps. Google Play is clearly the bigger market, representing the 72% of the total apps shared with nearly 4 million of apps, followed by Anzhi and Appchina with the 13% and 10% of the market, respectively.

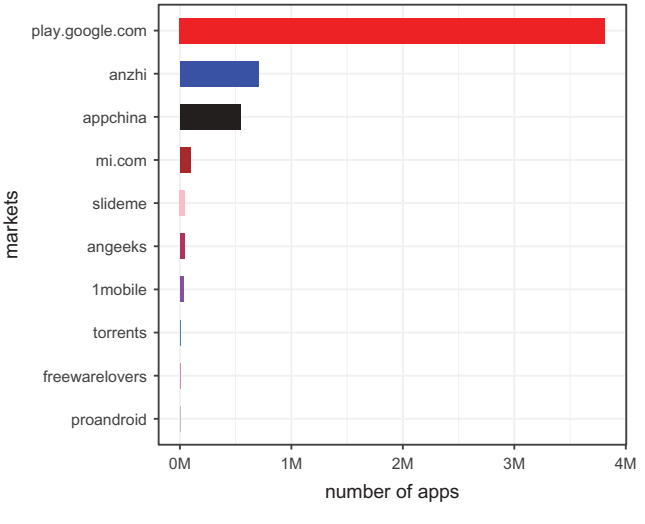


Figure 2. Distribution of the studied apps across the 10 most representative Android markets.

As a matter of fact, neither anti-malware is capable of detecting all existing malware, only a small subsample of the wide-known malware is recognized by a large number of anti-malware. With this as premise, we want to study the proportion of malicious apps detected in each market based on the scanning results of a large set of anti-malware vendors. This analysis allow us to depict the health status of Android markets, as well as the malware detection capabilities of various anti-malware vendors hosted by the VirusTotal web service.

Figure 3 illustrates the malware share in the 10 previously identified major Android markets. The y-axis represents the percent of apps that were flagged as malware by at least the number of vendors represented in the x-axis. According to the figure, Anzhi and AppChina contain the largest number of flagged apps. It is clear that anti-malware vendors have unequal scanning results. In fact, if we assume to flag an app as malware only if at least one anti-malware product has found it suspicious, then nearly the 20% of Google Play apps would be considered as malware. Hopefully, we see that these flagging rates drop steadily in all markets when we consider the scanning results of a larger set of anti-malware vendors (e.g. less than 2% of apps in Google Play are flagged as malware by more than 10 anti-malware).

Now, we focus our analysis in the diversity measures discussed previously (variety, ubiquity, balance and disparity). Regarding the first of these measures, Table 4 shows the variety of malware types and families in the 10 major markets. As expected, there is a strong correlation (Pearson coefficient = 0.98) between the variety of types and families in the markets. Google Play, Anzhi and Appchina are the three more diverse markets in terms of representation of different specimens. This result means that these markets should be considered as the most representative sources of real samples for performing malware research on the wild. By contrast, Proandroid, Freewarelovers and Torrents have the lower variety of malware, which suggest a lesser diversity.

Table 4 also presents the Rao-Stirling (*RS*) disparity values obtained for each market. In this case, the α and β parameters of Equation 4 were set to 0.1 and 0.9, respectively, in order to retain the most of the disparity, which allow to quantify the differences among malware regarding to their perceived behaviour for anti-malware engines. The results show that

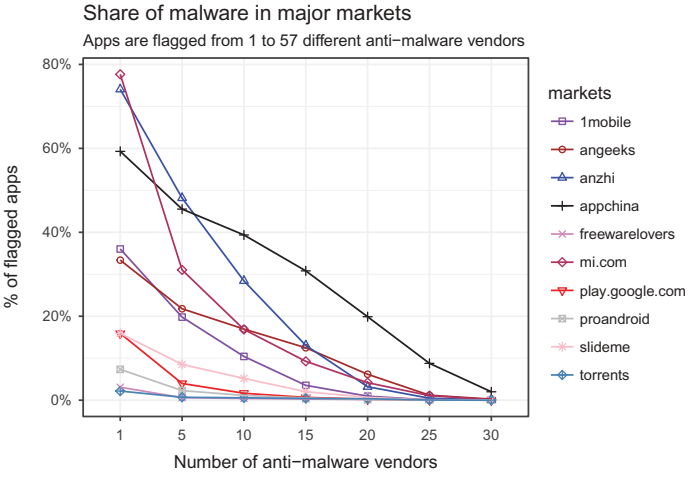


Figure 3. First 30 true malware detection rates in major Android markets according to the flagging results of 57 anti-malware vendors.

Table 4. Values of variety and disparity per types and families of malware in major Android markets.

	Variety		Disparity	
Market	Type	Family	Type	Family
Play.google.com	124	1110	17.08	628.20
Anzhi	97	1068	11.14	616.07
Appchina	71	723	7.12	431.09
mi.com	56	463	8.01	417.76
Slideme	23	72	2.29	29.22
Angeeks	29	196	4.08	162.71
1mobile	17	71	1.33	35.43
Torrents	7	12	1.92	8.75
Freewarelovers	6	11	1.18	6.48
Proandroid	5	19	0.63	9.66

Google Play and Anzhi share the higher malware disparity, while Appchina and Mi.com share similar values of disparity for types and families. The low disparity of the Slideme market is notable. In all cases, the values of the diversity measures for families are higher than those obtained for types because there are much more families than types in all markets.

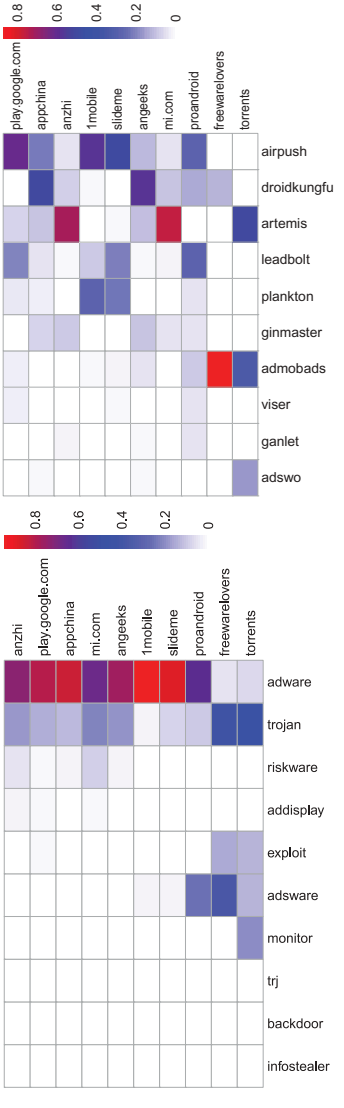
Table 5 presents an ordered list of the 10 more ubiquitous malware types and families for all studied markets. According to these results, the types Adware and Trojan are present in all markets, while the families ADMOBADS and DROIDKUNGFU are equally spread in almost all the markets. However, we believe that this perceived ubiquity could be doubtful because, as a general rule, when an anti-malware is not able to recognize the class of an Android malware, the assigned labels are usually the generic types ADWARE or TROJAN. From a diversity perspective, this result indicates that the families ADMOBADS and DROIDKUNGFU represent the most omnipresent threats, being present in 9 of the 10 major markets. This capacity to subsist in different environments could be due to the ability of these families to evolve in order to increase its level of sophistication, diversifying its features and exploiting new vulnerabilities.

We are interested in quantifying, for comparing purposes, how much of the most ubiquitous malware types and families a particular market has. Figure 4 depicts the proportions of malware, by types and families, which are present in each market. We can observe that ADWARE is a prevail type of malware in most markets, with the exception of Freewarelovers and Torrents that have a majority of TROJANS. On the other hand, it is notable that a large proportion of malware in the Freewarelovers market belongs to the Admobads family, while the Google Play market have a high proportion of malware belonging to the AIRPUSH family.

We use the Shannon Entropy measure in order to assess the evolution of malware regarding to the balance measure for a specific time interval. Figure 5 shows the balance

Table 5. Ubiquity of types and families of malware in major Android markets.

Type	Ubiquity	Family	Ubiquity
ADWARE	10	ADMOBADS	9
TROJAN	10	DROIDKUNGFU	9
ADSWARE	9	ADSWO	8
EXPLOIT	8	AIRPUSH	8
BACKDOOR	7	ARTEMIS	8
INFOSTEALER	7	GANLET	8
MONITOR	7	GINMASTER	8
RISKWARE	7	LEADBOLT	8
TRJ	7	PLANKTON	8
ADDISPLAY	6	VISER	8



(a) Types of malware. (b) Families of malware.

Figure 4. Pheatmaps of the proportions of malware per types and families that are present in major Android markets.

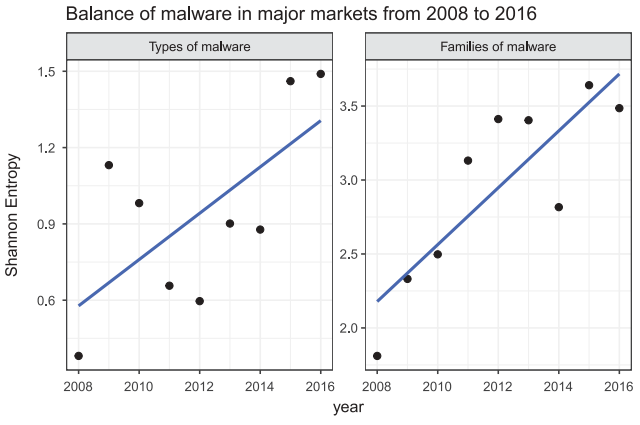


Figure 5. Balance of malware types and families according to its Shannon Entropy values since 2008.

of malware for all markets obtained in a period of 9 years, from 2008 to 2016. In general, we note a significant increase of balance for this period, specially from the year 2014, which make evident an increase in the diversity of the variants of malware since then. This result is important because it means that the malware is not only evolving in terms of complexity and specialization, but diversifying itself to new variants instead of concentrating its evolution into similar threats.

Figure 6 presents a treemap that shows the diversity of types and families of malware samples according to the labels assigned by the anti-malware engines. A treemap is a space-filling visualization of hierarchical structures able to take all the diversity dimensions into account [37]. In the figure, the size of the boxes is proportional to the number of malware assigned to each family, whereas the colours correspond to the main types of malware present in the data. Variety is represented by the number of boxes, balance is indicated by the differences in the size of the boxes and disparity is depicted by means of the different colours.

Regarding the variety, the treemap shows the presence of a large number of different variants of malware (a total of 2151 families distributed across 167 types). Regarding the

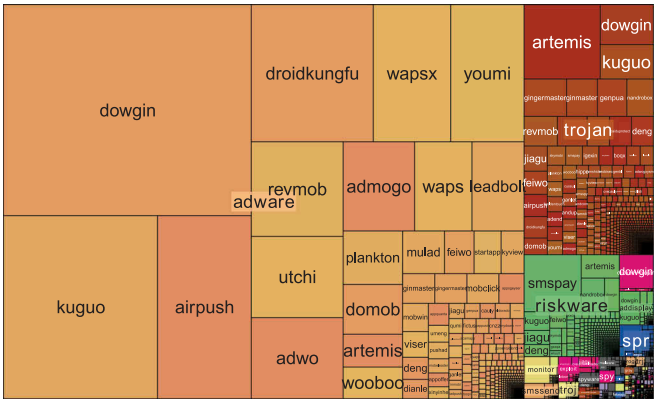


Figure 6. Treemap of malware types and family labels for all markets in the studied data set.

balance, we can observe that the malware is more concentrated (in terms of number of malicious apps) in the Adware category. The figure also shows that the higher levels of disparity occur in the less representative types of malware.

4.2. *Clustering classes of malware*

We perform an agglomerative hierarchical clustering analysis based on the labels assigned by the anti-malware to the apps. The clustering is conducted according to the types and families of malware. For this aim, we use the algorithm described in Section 2.3 and the dissimilarity measure proposed in Section 3.3. The results of this clustering offer insights about the relations among the labels assigned to the different varieties and specimens. It is a useful method to discover how the anti-malware perceives the relations among diverse classes of malware.

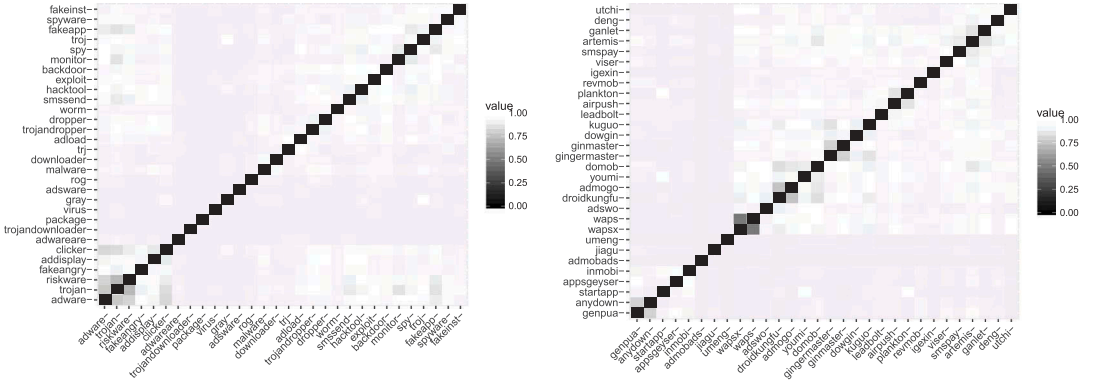
In this study, we consider as malware samples those apps that were flagged by more than 10 anti-malware. In order to facilitate the visualization of the results obtained, we reduce the number of malware samples for our clustering analysis. Accordingly, we randomly selected a subset of 100 malware samples for each of the 30 more ubiquitous types and families of malware. Hence, we build two datasets, the first consisting in 3000 labels of types of malware and the second consisting in 3000 labels of malware families.

We create a prototype of instance for each class of malware according to the anti-malware scanning reports. All the labels assigned to each class were joined to build a multiset of candidate labels for the classes. Figure 7 shows the initial dissimilarity matrices obtained for our datasets, which were calculated for the prototypes according to the dissimilarity measure proposed in Section 3.3. The dissimilarity matrix (a) shows a few square blocks of slightly correlated instances of the types of malware Adware, Trojan and Riskware, while in the dissimilarity matrix (b), it is notable the strong relation between the families Wapsx and Waps. In both matrices, the majority of pairwise values are very close to 1, showing a high dissimilarity among the classes of malware.

Figure 8 depicts the dendrograms derived from the application of the AGNES clustering method for both datasets. In the figure, each leaf corresponds to one malware label assigned. As we move up the tree, labels that are similar between each other are combined into branches, which are themselves fused at a higher height. The height of the fusion, provided on the vertical axis, indicates the dissimilarity between two classes, which can be interpreted as a diversification distance. The higher the height of the fusion, the more dissimilar the classes are.

Note that conclusions about the proximity of two classes can be drawn only based on the height where branches containing these two observations are firstly fused. Therefore, we cannot use the proximity of two malware labels along the horizontal axis as a criterion to measure their similarity.

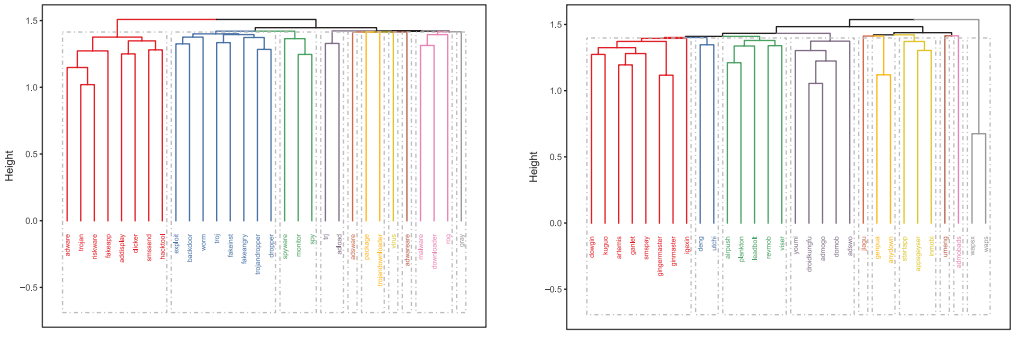
The dendrogram of malware types confirms the apparent labelling decision overlapping among the ADWARE, TROJAN and RISKWARE malware types. On the other hand, the dendrogram of malware families shows that WAPSX and WAPS are closely related; in other words, they seem to be variants of the same threat. The remaining families and types of malware seem to be rather unrelated, thus no significant labelling relations can be inferred from the clusters.



(a) Types of malware.

(b) Families of malware.

Figure 7. Dissimilarity matrix of pairwise correlation coefficients between types and families of malware.



(a) Types of malware. (b) Families of malware.

Figure 8. Dendrograms obtained after clustering the datasets of types and families of malware.

5. Discussion

Since the release of the Android operating system in 2008, malware targeting this platform has grown quickly in a short period of time; the future perspective is that malware continues proliferating with the increasing use of mobile devices. In this paper, we present empirical evidence that Android malware has not only increased, but also diversified significantly during the last years. The variety and ubiquity of malware becomes more evident at a large scale, being significant across both official and alternative markets. Our results on malware detection rates show that the majority of the malware is not simultaneously identified by several anti-malware technologies, which is in line with previous studies that discuss the weakness of signature-based detection systems. Only a small subset of common malware is detected unanimously by most anti-malware engines. This suggests that future detection systems should apply more advanced techniques, such as dynamic detection based on network traffic monitoring, in order to evaluate the trustworthiness of an app.

According to Zhou et al. [3], the scanning results based on the criteria of various anti-malware is able to show a dangerous malware infection rate for the Android markets studied in this paper. Considering a margin of true detection rate of 10 anti-malware engines, then the 2% of the total apps in the official Android market Google Play are at risk, whereas a mean of nearly 20% of apps could be considered as malware in other alternative markets. Such infection rates can compromise an enormous number of mobile phones and cause millionaire losses to companies and users, given the current widespread in the use of smartphones.

We note that the proportion of Adware and Trojan types of malware are significantly higher when compared to the rest of registered types. However, we have some concerns about these results as we note that when an anti-malware is not able to recognize the family of a suspicious app, then it chooses to assign it just the generics ADWARE or TROJAN labels. Hopefully, ADWARE malware is often not so harmful to the Android users (i.e. apps that continuously display undesired advertisement during its use). We also detect the presence of some pervasive TROJAN malware families such as DROIDKUNGFU, which is a dangerous malware that, in some of its more advanced versions, employs root exploits for obtaining root privileges, bypassing any built-in Android mechanism of security and compromising the user's personal information [6].

We collected two datasets for clustering analysis, both with 3000 apps flagged as malware by more than 10 anti-malware vendors, which we consider as representative malware samples. Our goal was to investigate the labelling results of the anti-malware engines. Based on an agglomerative hierarchical clustering analysis, we assessed the performance of 57 anti-malware to assigning types and family labels in order to distinguish among different malware behaviours. Our results support previous evidence that malware labelling is not a precise science, since each anti-malware follows its own mechanisms and heuristics to report a true detection [2].

One possible drawback of the detection approach based on multiple anti-malware is the inefficacy to correctly cluster a zero-day attack of a new malware specimen. However, due to the lack of a better solution at hand, it is still advisable the use of the diverse criteria of various anti-malware products for future research on automatic labelling of malicious apps.

The use of an ensemble of anti-malware vendors shows promising results in the area of malware identification. Our clustering results motivate further research on the aggregation of the decisions of anti-malware to improve the assignment of more accurate labels. There is a clear need to apply more rigorous detection analysis based on the scanning of anti-malware, in both official and unofficial Android markets, in order to accurately flagging suspicious apps on the wild. In this context, results are imperative to develop tools capable to quickly and automatically analyse large datasets of apps in order to reduce the number of suspicious apps to a small subset for further examination. We believe that ML techniques, such as Deep Neural Networks, represent an attractive approach to deal with this problem, as a viable alternative to flag malware variants more accurately through fast and scalable detection systems.

6. Related work

Previous research has focused on analysing the behaviour of Android malware from diverse points of view [2,6,38,39]. In particular, there is an increasing interest in studying the nature of malware in order to collect ground truths [27,40,41]. Researchers have reported the existence of a clear gap between the malware analysis results obtained in the lab and those obtained in the wild [28]. Accordingly, most of the work in Android malware is based on the use of advanced code analysis [9] and ML techniques [42] to detect malicious apps. For instance, Suarez-Tangil et al. [43] propose a clustering method that inspects code structures to analyse the evolution of families of Android malware, while Allix et al. [26] use various measures to beat the lack of consensus among anti-malware decisions and build ground truth datasets in the wild.

The use of ML techniques has demonstrated to be an effective approach for the automatic detection and classification of malicious Android apps [38,44,45]. ML methods comprise mostly the use of supervised classification methods [46,47] and clustering analysis [43,48]. Latest developed tools for detecting malware using ML techniques are based on the decisions given by a set of anti-malware engines. The primary role of an anti-malware engine is to decide whether a given sample should be considered as malicious. These decisions have important consequences in real-world environments, since a positive detection will probably trigger a response alert in order to mitigate the potential threat. False positives would thus lead to a waste of processing resources, while false negatives can have

direct consequences such as substantial system damages or data loses. Anti-malware engines must select an appropriate threshold between a reckless number of false positives and a harmful number of false negatives.

In spite of the promising results obtained, ML approaches based on multiple anti-malware decisions have not been widely implemented in the malware detection industry. In this sense, the validation of ML-based malware detection systems remains being a major issue. Many research works report that the validation in-lab of detection schemes is not a reliable indicator for real-world malware detectors. For example, Allix et al. [27] identify several parameters that can deceptively affect the performance of malware detectors, while Sommer and Paxson [49] highlight the detection differences appreciated between a closet lab environment and the wild real world. Therefore, it seems that there is a potential bias inherent to such results due to the evolving nature and diversification of malware. To mitigate this problem, there are various recommended practices to assess the quality of automated malware detection results using ML techniques. For more details on these issues, we refer the interested reader to [50] and [40].

7. Conclusions

We present an empirical study of Android malware from a diversity perspective. Our analysis comprises an extensive malware dataset, with more than 5 million of apps, which is based on a combination of the AndroZoo dataset and the Euphony tool. We use the scanning results of 57 anti-malware vendors to dissect four dimensions of malware diversity known as variety, ubiquity, balance and disparity. In addition to the diversity analysis presented and discussed in this paper, we explored the application of hierarchical clustering to get insights about the relations among different types and families of malware. Hence, we propose a dissimilarity measure based on multisets to group the most ubiquitous classes of malware using the labels given by anti-malware vendors. As was discussed in this paper, more efficient tools for automated malware detection and labelling, such as those based on ML techniques, are needed to cope with the increasing diversification of Android malware reported. From the results obtained through our diversity analysis and the clustering of malware labels, we can derive the following conclusions:

- There are empirical evidences that Android malware subsists in both official and alternative virtual markets, being significantly higher its presence in the latter.
- The types of malware ADWARE and TROJAN are ubiquitous in all studied markets, while ADMOBADS and DROIDKUNGFU are the most representative families detected.
- The balance of malware shows an increase during last years, which is an evidence of its continuous diversification.
- The markets Google Play and Anzhi share a similar malware disparity in spite of their different size, suggesting that the Anzhi market is the most diverse in terms of the malware retained.
- The use of the criteria of several anti-malware is a suitable approach to classify the malware in different types and families; no significant labelling conflicts among anti-malware were found in this study.

- The use of agglomerative hierarchical clustering, based on anti-malware labelling decisions, is a useful approach to group malware samples into families that perform a similar behaviour.

Notes

1. https://androzoo.uni.lu.
2. https://www.virustotal.com.
3. http://www.caro.org/articles/naming.html.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

César Soto-Valero received his MSc degree in Computer Science from the Universidad Central 'Marta Abreu' de las Villas, Cuba, in 2016. His research interests include data science, machine learning, software engineering, mining software repositories, time series analysis and knowledge-based systems.

Mabel González received her MSc degree in Computer Science from the Universidad Central 'Marta Abreu' de las Villas, Cuba, in 2010, and her PhD degree from the University of Granada, Spain, in 2016. She has published in prestigious journals such as Information Sciences (INS) and Knowledge and Information Systems (KAIS). Her research interests are mainly focused on data mining, semi-supervised learning and time-series classification methods.

ORCID

César Soto-Valero http://orcid.org/0000-0003-0541-6411
Mabel González http://orcid.org/0000-0003-0152-444X

References

1. Statista. Installed base of smartphones by operating system from 2015 to 2017. 2018. [cited Jan 16] Available from: https://www.statista.com/statistics/385001/smartphone-worldwide-installed-base-operating-systems/.
2. Allix K, Jerome Q, Bissyandé TF, et al. A forensic analysis of android malware – how is malware written and how it could be detected? In: Proceedings of the 2014 IEEE 38th Annual Computer Software and Applications Conference; Washington, DC, USA. IEEE Computer Society; 2014. p. 384–393; COMPSAC '14. Available from: DOI:10.1109/COMPSAC.2014.61.
3. Zhou Y, Wang Z, Zhou W, et al. Hey, you, get off of my market: detecting malicious apps in official and alternative android markets. In: NDSS; Vol. 25; 2012. p. 50–52.
4. Chin E, Felt AP, Sekar V, et al. Measuring user confidence in smartphone security and privacy. In: Proceedings of the Eighth Symposium on Usable Privacy and Security; New York, NY, USA. ACM; 2012. p. 1: 1–1:16;SOUPS '12. Available from: DOI:10.1145/2335356.2335358.
5. Zhou W, Zhou Y, Jiang X, et al. Detecting repackaged smartphone applications in third-party android marketplaces. In: Proceedings of the Second ACM Conference on Data and Application Security and Privacy; New York, NY, USA. ACM; 2012. p. 317–326; CODASPY '12. Available from: DOI:10.1145/2133601.2133640.

6. Jiang X, Zhou Y. Android malware. New York: Springer-Verlag; 2013.
7. Enck W A study of android application security. Proceedings of the 20th USENIX Security. 2011.
8. Felt AP, Chin E, Hanna S, et al. Android permissions demystified. In: Proceedings of the 18th ACM Conference on Computer and Communications Security; New York, NY, USA. ACM; 2011. p. 627–638; CCS '11. Available from: DOI:10.1145/2046707.2046779.
9. Yan LK, Droidscope: YH Seamlessly reconstructing the os and dalvik semantic views for dynamic android malware analysis. In: USENIX security symposium; 2012. p. 569–584.
10. Kalaimannan E, John SK, DuBose T, et al. Influences on ransomware's evolution and predictions for the future challenges. J Cyber Security Technol. 2016. DOI:10.1080/23742917.2016.1252191.
11. Maggi F, Bellini A, Salvaneschi G, et al. Finding non-trivial malware naming inconsistencies. In: ICISS; Springer; 2011. p. 144–159.
12. Baudry B, Monperrus M, Mony C, et al. Diversify: ecology-inspired software evolution for diversity emergence. In: 2014 Software Evolution Week – IEEE Conference on Software Maintenance, Reengineering, and Reverse Engineering (CSMR-WCRE); Feb; 2014. p. 395–398.
13. Borello JM, É F, Mé L. From the design of a generic metamorphic engine to a black-box classification of antivirus detection techniques. J Comput Virol. 2010 Aug 6;(3):277–287. Available from. DOI:10.1007/s11416-009-0136-2
14. Baudry B, Monperrus M. The multiple facets of software diversity: recent developments in year 2000 and beyond. ACM Comput Surv. 2015 Sep;48(1):1–16. Available from.
15. Mendez D, Baudry B, Monperrus M Empirical evidence of large-scale diversity in API usage of object-oriented software. In: 2013 IEEE 13th International Working Conference on Source Code Analysis and Manipulation (SCAM); Sept; 2013. p. 43–52. DOI:10.1109/SCAM.2013.6648183.
16. Stirling A. A general framework for analysing diversity in science, technology and society. J Royal Soc Interface. 2007;4(15):707–719.
17. Zhou Y, Jiang X Dissecting android malware: characterization and evolution. In: IEEE Symposium on Security and Privacy (SP); 2012. p. 95–109.
18. Sebastián M, Rivera R, Kotzias P, et al. Avclass: A tool for massive malware labeling. Cham: Springer International Publishing; 2016. 230–253. DOI:10.1007/978-3-319-45719-2_11
19. Hurier M, Suarez-Tangil G, Dash SK, et al. Euphony: harmonious unification of cacophonous anti-virus vendor labels for android malware. In: Proceedings of the 14th International Conference on Mining Software Repositories; Piscataway, NJ, USA. IEEE Press; 2017. p. 425–435; MSR '17. Available from: DOI:10.1109/MSR.2017.57.
20. Li P, Liu L, Gao D, et al. On challenges in evaluating malware clustering. In: RAID; Vol. 6307; Springer; 2010. p. 238–255.
21. Rafique MZ, Firma: CJ. Malware clustering and network signature generation with mixed network behaviors. Berlin, Heidelberg: Springer Berlin Heidelberg; 2013. 144–163. DOI:10.1007/978-3-642-41284-4_8
22. Crussell J, Gibler C, Chen H Attack of the clones: detecting cloned applications on android markets. In: ESORICS; Vol. 12; Springer; 2012. p. 37–54.
23. Bayer U, Comparetti PM, Hlauschek C, et al. Scalable, behavior-based malware clustering. In: NDSS; Vol. 9; 2009. p. 8–11.
24. Jang J, Brumley D, Venkataraman S. Bitshred: feature hashing malware for scalable triage and semantic analysis. In: Proceedings of the 18th ACM Conference on Computer and Communications Security; New York, NY, USA. ACM; 2011. p. 309–320; CCS '11. DOI:10.1145/2046707.2046742.
25. Rousseeuw PJ, Kaufman L. Finding groups in data. Hoboken: Wiley Online Library; 1990.
26. Allix K, Bissyandé TF, Klein J, et al. Androzoo: collecting millions of android apps for the research community. In: 2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR); May; 2016. p. 468–471.
27. Allix K, Bissyandé TF, Jérome Q, et al. Empirical assessment of machine learning-based malware detectors for android. Empirical Softw Eng. 2016 Feb;21(1):183–211. Available from.
28. Allix K, Bissyandé TF, Klein J, et al. Are your training datasets yet relevant. In: International Symposium on Engineering Secure Software and Systems; Springer; 2015. p. 51–67.

29. Hecht G, Benomar O, Rouvoy R, et al. Tracking the software quality of android applications along their evolution. In: 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE); Nov; 2015. p. 236–247.
30. Li L, Bartel A, Bissyandé TF, et al. Iccta: detecting inter-component privacy leaks in android apps. In: Proceedings of the 37th International Conference on Software Engineering – Volume 1; Piscataway, NJ, USA. IEEE Press; 2015. p. 280–291; ICSE '15. Available from: http://dl.acm.org/citation.cfm?id=2818754.2818791.
31. Guevara MR, Hartmann D, Mendoza M. Diverse: an r package to analyze diversity in complex systems. R J. 2016;8(2):60–78.
32. Tuomisto H. An updated consumer's guide to evenness and related indices. Oikos. 2012;121 (8):1203–1218.
33. Shannon CE. A mathematical theory of communication. SIGMOBILE Mob Comput Commun Rev. 2001 Jan 5;(1):3–55. DOI:10.1145/584091.584093
34. Wang J, Thijs B, Glänzel W. Interdisciplinarity and impact: distinct effects of variety, balance, and disparity. Plos One. 2015 05;10(5):1–18.
35. Kosters WA, Laros JFJ. Metrics for mining multisets. In: Bramer M, Coenen F, Petridis M, editors. Research and development in intelligent systems XXIV. London: Springer London; 2008. p. 293–303. DOI:10.1007/978-1-84800-094-0_22.
36. Deza MM, Deza E. Encyclopedia of distances. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. 1–583. DOI:10.1007/978-3-642-00234-2_1
37. Bederson BB, Shneiderman B, Wattenberg M. Ordered and quantum treemaps: making effective use of 2D space to display hierarchies. ACM Trans Graph. 2002 Oct;21(4):833–854.
38. Wu DJ, Mao CH, Wei TE, et al. Droidmat: android malware detection through manifest and API calls tracing. In: 2012 Seventh Asia Joint Conference on Information Security; Aug; 2012. p. 62–69.
39. Burguera I, Zurutuza U, Nadjm-Tehrani S. Crowdroid: behavior-based malware detection system for android. In: Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices; New York, NY, USA. ACM; 2011. p. 15–26; SPSM '11. DOI:10.1145/2046614.2046619.
40. Hurier M, Allix K, Bissyandé TF, et al. On the lack of consensus in anti-virus decisions: metrics and insights on building ground truths of android malware. Cham: Springer International Publishing; 2016. 142–162. DOI:10.1007/978-3-319-40667-1_8
41. Felt AP, Finifter M, Chin E, et al. A survey of mobile malware in the wild. In: Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices; New York, NY, USA. ACM; 2011. p. 3–14; SPSM '11. DOI:10.1145/2046614.2046618.
42. Chen W, Aspinall D, Gordon AD, et al. More semantics more robust: improving android malware classifiers. In: Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks; New York, NY, USA. ACM; 2016. p. 147–158; WiSec '16. DOI:10.1145/2939918.2939931.
43. Suarez-Tangil G, Tapiador JE, Peris-Lopez P, et al. Dendroid: a text mining approach to analyzing and classifying code structures in android malware families. Expert Syst Appl. 2014;41(4):1104–1117. Available from: http://www.sciencedirect.com/science/article/pii/S0957417413006088
44. Aafer Y, Du W, Yin H. Droidapiminer: mining API-level features for robust malware detection in android. Cham: Springer International Publishing; 2013. 86–103. Available from. DOI:10.1007/978-3-319-04283-1_6
45. Chakradeo S, Reaves B, Traynor P, et al. Mast: triage for market-scale mobile malware analysis. In: Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks; New York, NY, USA. ACM; 2013. p. 13–24; WiSec '13. DOI:10.1145/2462096.2462100.
46. Narudin FA, Feizollah A, Anuar NB, et al. Evaluation of machine learning classifiers for mobile malware detection. Soft Comput. 2016 Jan;20(1):343–357.
47. Peng H, Gates C, Sarma B, et al. Using probabilistic generative models for ranking risks of android apps. In: Proceedings of the 2012 ACM Conference on Computer and

Communications Security; New York, NY, USA. ACM; 2012. p. 241–252; CCS '12. DOI: 10.1145/2382196.2382224.
48. Perdisci R, Vamo UM: Towards a fully automated malware clustering validity analysis. In: Proceedings of the 28th Annual Computer Security Applications Conference; New York, NY, USA. ACM; 2012. p. 329–338; ACSAC '12. DOI: 10.1145/2420950.2420999.
49. Sommer R, Paxson V Outside the closed world: on using machine learning for network intrusion detection. In: 2010 IEEE Symposium on Security and Privacy; May; 2010. p. 305–316.
50. Rossow C, Dietrich CJ, Grier C, et al. Prudent practices for designing malware experiments: status quo and outlook. In: 2012 IEEE Symposium on Security and Privacy; May; 2012. p. 65–79.