AV-Meter: An Evaluation of Antivirus Scans and Labels

Aziz Mohaisen[1] and Omar Alrawi[2]

[1]Verisign Labs [2]Qatar Computing Research Institute

Abstract. Antivirus scanners are designed to detect malware and, to a lesser extent, to label detections based on a family association. The labeling provided by AV vendors has many applications such as guiding efforts of disinfection and countermeasures, intelligence gathering, and attack attribution, among others. Furthermore, researchers rely on AV labels to establish a baseline of ground truth to compare their detection and classification algorithms. This is done despite many papers pointing out the subtle problem of relying on AV labels. However, the literature lacks any systematic study on validating the performance of antivirus scanners, and the reliability of those labels or detection.

In this paper, we set out to answer several questions concerning the detection rate, correctness of labels, and consistency of detection of AV scanners. Equipped with more than 12,000 malware samples of 11 malware families that are manually inspected and labeled, we pose the following questions. How do antivirus vendors perform relatively on them? How correct are the labels given by those vendors? How consistent are antivirus vendors among each other? We answer those questions unveiling many interesting results, and invite the community to challenge assumptions about relying on antivirus scans and labels as a ground truth for malware analysis and classification. Finally, we stress several research directions that may help addressing the problem.

Key words: Malware, Labeling, Automatic Analysis, Evaluation.

1 Introduction

Antivirus (AV) companies continuously evolve to improve their products, which protect users and businesses from malicious software (malware) threats. AV products provide two major functionalities: detection, the main focus of many AV companies, and labeling, a by-product of the detection with many important applications [27]. Labeling is an important feature to various parties: AV vendors, information security professionals, and the academic community. Labeling allows AV vendors to filter known malware and focus on new malware families or variants of familiar families with known remedies, and enables AV vendors to track a malware family and its evolution—thus allowing them to proactively create and deploy disinfection mechanisms of emerging threats [25]. In security operations, which are done in many enterprises, information security practitioners use malware labels to mitigate the attacks against their organization by deploying the proper disinfection mechanisms and providing the related risk assessment. Law enforcement agencies rely on labels for attack attribution. Finally, researchers have benefited from detection and labeling of malware provided by AV vendors in establishing baselines to compare their malware analysis and classification designs against [6,7,16,30,32,35,39,42].

► **Antivirus Labeling and Inconsistency.** The AV market is very diverse and provides much room for competition, allowing vendors to compete for a share of the market [28]. Despite various benefits [11], the diversity of AV software vendors creates a lot of disorganization due to the lack of standards and (incentives for) information sharing, malware family naming, and transparency. Each AV company has its own way of naming malware families [20]. Analysts, who study new malware samples, by utilizing artifacts within the malware to derive and give them names, usually create Malware names. Some malware families are so popular in underground forums, like SpyEye [23], Zeus [17], ZeroAccess [1], DirtJumper [5], etc., and AV vendors use those names given in the underground market. Other smaller and less prominent families are named independently by each AV company. For example, targeted malware [38]—stealthy and less popular—is tracked independently by AV vendors resulting in different naming.

The diversity of the market with the multi-stakeholder model is not the only cause of labeling problems. The problems can happen within the same vendor when an engine detects the same malware family with more than one label due to evasion techniques and evolution patterns over time. For example, a malware could be detected using a static signature, then detected later heuristically using a generic malicious behavior (due to polymorphism). In such case, the AV vendor will give it another label creating inconsistency within the labeling schema. These inconsistencies and shortcomings may impact applications that use AV labeling.

► **Inconsistencies Create Inefficiencies.** In light of the shortcomings highlighted above, the use of AV labels for validating malware classification research has some pitfalls. Malware samples collected by researchers are often not represented in their entirety within a single malware scanning engine. Accordingly, researchers are forced to use multiple engines to cover their datasets, thus forced to deal with inconsistencies in labeling and naming conventions. Researchers resolve the inconsistencies by translating names used across various vendors. However, given that different AV vendors may use different names to mean and refer to the same family, this translation effort is never easy nor complete. Even worse, different families may have the same name in different AV detections—for example "generic" and "trojan" are used by many vendors as an umbrella to label [25], sometimes making such translation impossible.

Furthermore, the detection and labeling inconsistencies create inefficiencies in the industry. For example, if a user of an AV engine detects a malware with a certain label, the user might have a mitigation plan for that malware family. On the other hand, another AV vendor may detect the same malware and give it a different label that is unfamiliar to the user, thus the user will not be able to use an existing mitigation plan for the same malware. This inefficiency can cost organizations a lot (directly or indirectly) and damage their reputation. While companies are secretive on that matter, some recent incidents include highlight the cost of compromise [14,26,36].

► **An "Elephant in the Room".** Sadly, while we are not the first to observe those inefficiencies in AV labeling systems [6, 7, 34], the community so far spent so little time systematically understanding them, let alone quantifying the inefficiencies and providing solutions to address them. Some of the work that pointed out the problem with AV labels used the same labels for validating algorithms by establishing a ground truth and a baseline [7, 34]. A great setback to the community's effort in pursuing this obvious and crucial problem is the lack of a better ground-truth than that provided by the AV scanners, a limitation we address in this work by relying on more than $12,000$ highly-accurate and manually vetted malware samples (more details in §3.1). We obtain those samples from real-world information security operations (§3.2), where vetting and highly accurate techniques for malware family labeling are employed as a service.

In this work we are motivated by the lack of a systematic study on understanding the inefficiencies of AV scanners for malware labeling and detections. Previous studies on the topic are sketchy, and are motivated by the need of making sense of provided labels to malware samples [31], but not testing the correctness of those labels or the completeness of the detections provided by different scanners. Accordingly, we develop metrics to evaluate the completeness, correctness, consistency, and coverage (defined in §2), and use them to evaluate the performance of various scanners. Our measurement study does not trigger active scans, but rather depends on querying the historical detections provided by each AV engine. While AV scanners' first priority is a high detection rate, we show that several scanners have low detection rates on our dataset. We show those findings by demonstrating that any sample we test exists in at least one AV scanner, thus one can obtain full detection of the tested samples using multiple vendors.

► **Contribution.** The contribution of this study is twofold. We provide metrics for evaluating AV detections and labeling systems. Second, we use manually vetted dataset for evaluating the detections and labeling of large number of AV engines using the proposed metrics. As a complementary contribution, we emphasize several research directions to address the issues raised in this study. To the best of our knowledge, there is no prior systematic work that explores this direction at the same level of rigor we follow in this paper (for the related work, see §6). Notice that we disclaim any novelty in pointing out the problem. In fact, there has been several works that pointed out problems with AV labels [6, 7], however those works did not systematically and quantitatively study the performance of AV scanners and the accuracy of their labels. This, as mentioned before, is in part because of the lack of datasets with solid ground truth of their label.

► **Shortcomings.** Our study has many shortcomings, and does not try to answer many questions that are either out of its scope or beyond our resources and capabilities. First of all, our study cannot be used as a generalization on how AV vendors would perform against each other in other contexts, because we do not use every sample in every given AV scanner. Similarly, the same generalization cannot be used over malware families, since we did not use all samples known by the AV scanners. Our study is, however, meaningful in answering the context's questions it poses for 12,000 malware samples that belong to various families. Furthermore, our study goes beyond the best known work in the literature on the problem by not relying on AV-provided vendors as reference for comparing other vendors (further details are in §6).

Another shortcoming of our study is the representation of families and their diversity. Families we studied fall under three classes: commercial DDoS, targeted, and mass-market families. While we believe that the 11 families we studied are fairly large to draw some conclusions, they are not representative to the large population of thousands of families a typical AV vendor would have, and conclusions cannot be gener-

alized. For example, our study does not consider classification of "nuisanceware", yet another class of scam malware via unethical marketing techniques. AV scanners are shown in the literature to perform worse for this class of malware [19], and one may deduce that this class would have also a worse classification and labeling rates than other families, although we were not able to concretely show that for the lack of data.
▶ **Organization.** The organization of the rest of this paper is as follows. In section 2 we review several metrics for the evaluation of AV scanners. In section 3 we provide an overview of the dataset we used in this study and the method we use for obtaining it. In section 4 we review the measurements and findings of this study. In section 5 we discuss implications of the findings and remedies, emphasizing several open directions for investigation. In section 6 we review the related work, followed by concluding remarks and the future work in section 7.

2 Evaluation Metrics

For formalizing the evaluation of the AV scanners, we assume a reference dataset $\mathcal{D}_i$ (where $1 \leq i \leq \Omega$ for Ω tested datasets). $\mathcal{D}_i$ consists of Δ_i samples of the same ground-truth label ℓ_i. We assume a set of scanners $\mathcal{A}$ of size Σ. Furthermore, we assume that each scanner (namely, a_j in $\mathcal{A}$ where $1 \leq j \leq \Sigma$) is capable of providing detection results for $\Delta'_{ij} \leq \Delta_i$ samples, denoted as $\mathcal{S}'_{ij} \subseteq \mathcal{D}_i$ (collectively denoted as $\mathcal{S}'_i$). Among those detections, we assume that the scanner a_j is capable of correctly labeling $\Delta''_{ij} \leq \Delta'_{ij}$ samples with the label ℓ_i. We denote those correctly labeled samples by a_j as $\mathcal{S}''_{ij} \subseteq \mathcal{S}'_{ij}$ (collectively denoted as $\mathcal{S}''_j$). In this work we use several evaluation metrics: the completeness, correctness, consistency, and coverage, which we define as follows.
▶ **Completeness.** For a given reference dataset, we compute the *completeness* score (commonly known as *detection rate*) of an AV scanner as the number detections returned by the scanner normalized by the size of the dataset. This is, for $\mathcal{D}_i$, a_j, Δ_i, and Δ'_{ij} that we defined earlier, we compute the completeness score as Δ'_{ij}/Δ_i.
▶ **Correctness.** For a given reference dataset, we compute the *correctness* score of a scanner as the number of detections returned by the scanner with the correct label as the reference dataset normalized by the size of the dataset. This is, for $\mathcal{D}_i$, a_j, Δ_i, and Δ''_{ij} we defined earlier, we compute the correctness score as Δ''_{ij}/Δ_i.
▶ **Consistency.** The *consistency* measures the extent to which different scanners agree in their detection and labeling of malware samples. As such, we define two versions of the score, depending on the metric used for inclusion of samples: completeness or correctness. We use the Jaccard index to measure this agreement in both cases. For the completeness-based consistency, the consistency is defined as the size of the intersection normalized by the size of the union of sample sets detected by both of the two scanners. Using the notation we defined above, and without losing generality, we define the completeness-based consistency of a_j and a_r as $|\mathcal{S}'_{ij} \cap \mathcal{S}'_{ir}|/|\mathcal{S}'_{ij} \cup \mathcal{S}'_{ir}|$. Similarly, we define the correctness-based consistency as $|\mathcal{S}''_{ij} \cap \mathcal{S}''_{ir}|/|\mathcal{S}''_{ij} \cup \mathcal{S}''_{ir}|$.
▶ **Coverage.** We define the coverage as the minimal number of AV scanners that we need to utilize so that the size of the detected (or correctly labeled) samples is maximal. Alternatively, we view the coverage for a number of AV scanners as the maximal ratio of collectively detected (or correctly labeled) samples by those scanners normalized by

the total number of samples scanned by them. Ideally, we want to find the minimal number of scanners k, where $\mathcal{A}_k = \{a_1, \ldots, a_k\}$, which we need to use so that the completeness (or the correctness) score is 1. This is done by repetitively selecting the AV scanner that has the most number of samples not included so far in the result until all samples are covered.

Related to both completeness and correctness scores are the number of labels provided by each AV scanner, and the number of malware samples labeled under the largest label. Indeed, one can even extend the latter metric to include the distribution on the size of all labels provided by an AV scanner for each malware family. We compute those derived metrics for each scanner, label, and malware family.

3 Datasets, Labels, and Scans

3.1 Dataset

For the evaluation of different AV vendors based on a common ground of comparison, we use a multitude of malware samples. Namely, we use more than 12,000 malware samples that belong to 11 distinct malware families. Those families include targeted malware, which are oftentimes low-key and less populated in antivirus scanners, DDoS malware, rootkits, and trojans that are more popular and well populated in antivirus scanners and repositories. We use families, such as Zeus, with leaked codes that are well understood in the industry. The malware families used in the study are shown in Table 1 with the number of samples that belong to each malware family, and the corresponding brief description. Finally, we emphasize that our dataset contains only malware, and no benign binaries, thus we do not study false positives for detection in the rest of this work. In the following, we elaborate on each of those families.

- **Zeus:** Zeus is a banking Trojan that targets the financial sector by stealing credentials from infected victims. The malware steals credentials by hooking Windows API functions which intercepts communication between clients and bank's website and modifies the returning results to hide its activities.
- **Avzhan:** is a DDoS botnet, reported by Arbor Networks in their DDoS and security reports in September 2010 [3]. The family is closely related to the IMDDoS [9], a Chinese process-based botnet announced by Damballa around September 2010. Similar to IMDDoS, Avzhan is used as a commercial botnet that can be hired (as a hit man) to launch DDoS attacks against targets of interest. The owners of the botnet claim on their website that the botnet can be used only against non-legitimate websites, such as gambling sites.
- **Darkness:** (Optima) is available commercially and developed by Russian criminals to launch DDoS, steal credentials and use infected hosts for launching traffic tunneling attacks (uses infected zombies as potential proxy servers). The original botnet was released in 2009, and as of end of 2011 it is in the 10th generation [10].
- **DDoSer:** Ddoser, also know as Blackenergy, is a DDoS malware that is capable of carrying out HTTP DDoS attacks. This malware can target more than 1 IP address per DNS record, which makes it different than the other DDoS tools. It was reported on by Arbor networks and analyzed in 2007 [12].

- **JKDDoS**, a DDoS malware family that is targeted towards the mining industry [4]. The first generation of the malware family was observed as early as September of 2009, and was reported first by Arbor DDoS and security reports in March 2011.
- **N0ise:** a DDoS tool with extra functionalities like stealing credentials and downloading and executing other malware. The main use of n0ise is recruiting other bots to DDoS a victim using methods like HTTP, UDP, and ICMP flood [21].
- **ShadyRat:** used to steal sensitive information like trade secrets, patent technologies, and internal documents. The malware employs a stealthy technique when communicating with the C2 by using a combination of encrypted HTML comments in compromised pages or steganography in images uploaded to a website [22]
- **DNSCalc:** is a targeted malware that uses responses from the DNS request to calculate the IP address and port number it should communicate on, hence the name DNSCalc. The malware steals sensitive information and targets research sector [13].
- **Lurid:** a targeted malware family, where three hundred attacks launched by this malware family were targeted towards 1465 victims, and were persistent via monitoring using 15 domain names and 10 active IP addresses. While the attacks are targeted towards US government and non-government organization (NGOs), there seems to be no relationship between the targets indicating its commercial use [40]
- **Getkys:** (Sykipot) is a single-stage Trojan that runs and injects itself into three targeted processes: outlook.exe, iexplorer.exe and firefox.exe. Getkys communicates via HTTP requests and uses two unique and identifiable URL formats like the string "getkys." The malware targets aerospace, defense, and think tank organizations [2].
- **ZAccess:** also known as ZeroAccess, is a rootkit-based Trojan and is mainly used as an enabler for other malicious activities on the infected hosts (following a pay-per-click advertising model). It can be used to download other malware samples, open backdoor on the infected hosts, etc. The family was reported by Symantec in July 2011, and infects most versions on the windows operating system [1]

3.2 Samples Analysis, Vetting, and Labeling

Analysts have identified each malware sample in our dataset manually over a period of time in a service that requires reverse engineering and manual assignment and vetting of the assigned labels. Our dataset consists of variety of families and a large number of total samples, which enables us to derive meaningful insights into the problem at hand. Furthermore, compared to the prior literature that relies on tens to hundreds of thousands of malware samples, our dataset is small enough to enable manual vetting and manual label assignment. For the data we use in the study, we use malware samples accumulated over a period of 18 months (mid 2011 to 2013). This gives the AV vendors an advantage and might overestimate their performance compared to more emerging or advanced persistent threat (APT)—or greyware/spyware, where AV vendors are known to perform worse [19].

To identify the family to which a malware sample belongs, an analyst runs the malware sample through static analysis, dynamic analysis, and context (customer feedback) analysis. For the static analysis, artifacts like file name, size, hashes, magic literals, compression artifacts, date, source, author, file type, portable executable (PE) header,

Table 1. Malware families used in this study, their size, and description. All scans done on those malware samples are in May 2013. (t) stands for targeted malware families. Ddoser is also known as BlackEnergy while Darkness is known as Optima.

Malware family	#	description
Avzhan	3458	Commercial DDoS bot
Darkness	1878	Commercial DDoS bot
Ddoser	502	Commercial DDoS bot
Jkddos	333	Commercial DDoS Bot
N0ise	431	Commercial DDoS Bot
ShadyRAT	1287	(t) targeted gov and corps
DNSCalc	403	(t) targeted US defense companies
Lurid	399	(t) initially targeted NGOs
Getkys	953	(t) targets medical sector
ZeroAccess	568	Rootkit, monetized by click-fraud
Zeus	1975	Banking, targets credentials

sections, imports, import hash, and resources, as well as compiler artifacts, are used. For the dynamic analysis, we run the sample in a virtual environment (or on the bare metal if needed) and collect indicators like file system artifacts, memory artifacts, registry artifacts, and network artifacts—more details on those artifacts and indicators are in [24] and [41]. An analyst based on the collective nature of those indicators, and by utilizing customer input and private security community consensus and memory signatures, provides labeling. For naming, we use what is collectively accepted in the AV community of names on samples that exhibit the behavior and use those indicators. For the evaluation of our data set we used VirusTotal signatures for 48 AV engines to test several evaluation measures. We discarded all engines that provided scans for less than 10% of our dataset.

Given that malware samples are not labeled using the same convention by all AV vendors and scanners, we rely on experts knowledge of the samples and the names given by those vendors to identify a common ground for names. In total, we used industry, community, and malware author given labels as correct labels for each malware family (details are in §4). The only exception was the targeted malware, for which we used labels given by the AV community. For example, zeus is often time named zbot in the industry, and is given a hierarchical suffix that indicates generational differences (or sample sequencing using signature-based techniques; e.g., zbot!gen[0-72] given by Symantec using heuristics). For that, we get rid of the suffix, and only use the stem of the name to unify the multitude of names given by the same vendor for various samples. Similarly, we utilize a similar technique for across-vendor label unification. When a family is called different names by different vendors (e.g., DNSCalc is named cosmu and ldpinch by different vendors), we use both names as a correct label.

Note that DDoS is not overrepresented in our data set, but the families represented belonged to the most accurately vetted ones. We have several other sets but we did not use them in this study because they did not have well known community labels that we can map to AV labels, hence they were left out. For those families and samples we lifted out, and by looking at the labels from AV, they did not converge on a clear label that we could use, and instead they resulted mostly in generic and heuristic labels.

3.3 VirusTotal

VirusTotal is a multi-engine AV scanner that accepts submissions by users and scans the sample with those engines. The results from VirusTotal have much useful information, but for our case we only use the AV vendor name and their detection label. VirusTotal will provide more AV results (with respect to both the quantity and quality) when a malware sample has been submitted in the past. The reason for this is that AV engines will provide an updated signature for malware that is not previously detected by their engines but was detected by other engines. Hence, malware samples that have been submitted multiple times for a long period of time will have better detection rates, and labels given to them by AV vendors are likely to be consistent, correct, and complete. We run our dataset through VirusTotal and obtain detections and labels of the detections for every sample. We use the most recent detection and label given by VirusTotal.

Finally, we emphasize the difference between vendor and scanner, since some vendors have multiple scanners—as a result of multiple products—in VirusTotal. For example, we note that NOD32 and McAfee have two scanners in the reported results. When there is more than one scanner per vendor, we use the one with the highest results to report on the performance on that vendor. We also emphasize the method described in section 3.2 for identifying malware samples by a family name.

4 Measurements and Findings

4.1 Completeness (Detection Rate)

For completeness, and as explained above, we use the ratio of detections for every AV scanner and for each of the families studied (the ratio is computed over the total number of malware samples in each family). For example, an AV engine $\mathcal{A}_i$ that has 950 detections out of a 1,000 sample dataset would have a 0.95 completeness regardless to what labels that are returned by the named AV.

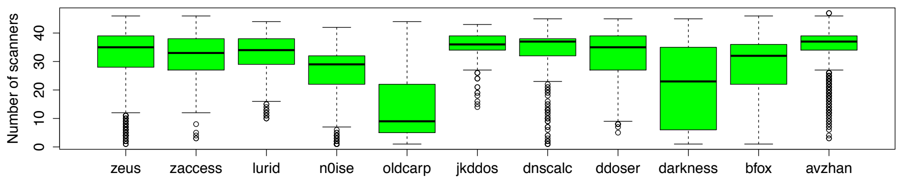

Fig. 1. Number of scanners that detected each sample in our dataset grouped by family.

► **Samples populated in scanners.** We consider the number of AV scanners that detect each sample, and group them by family. Figure 1 shows the various families used in this paper, and a box plot for the number of the scanners that detected each sample in each family. From this figure we observe that with the exception of two families (darkness and oldcarp; aka Getkys), the majority of samples are detected by more than half of the scanners. Furthermore, in relation with the rest of figures in this section, this figure shows that the majority of families contribute to the high detection rate.
► **Overall completeness scores.** Figure 2 shows the completeness scores of each of the AV scanners listed on the x-axis, for the 11 families in Table 1. Each of the boxes in the boxplot corresponds to the completeness distribution of the given scanner: the median of the completeness for the AV scanner over the 11 families is marked as the thick middle line, the edges of the box are the first and third quartiles, and the boundaries of each plot are the minimum and maximum with the outliers below 5% and above 95% of the population distribution. On this figure, we make the following remarks and findings. First of all, we notice that the maximum completeness provided by any AV scanner for any of the studied malware families is 0.997 (99.7% detection rate). We later show that all samples are present in a set of independent scanners, when considered combined, suggesting that those malware samples are not obsolete or limited or present only in our malware repository. Second, we note that on average the completeness of the scanners with respect to the total number of malware families considered in the study is only 0.591 (a score not shown in the figure; which means only 59.1% detection rate). Furthermore, the same figure shows that even with the well performing scanners on

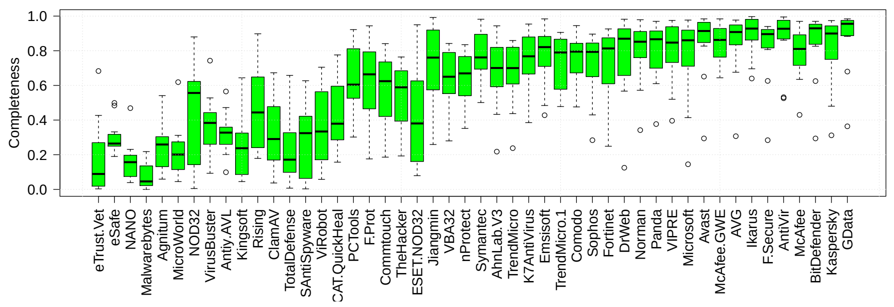

Fig. 2. A box plot of the completeness scores of antivirus scanners used in the study against the 11 malware families shown in Table 1. The y-axis is on the linear scale, of 0-1.

the majority of samples and families, there are always families that are missed by the majority of scanners—Darkness and Oldcarp in Figure 1, and are statistically considered outliers with respect to the rest of the scores provided by the same scanners for other families (e.g., scanners on the right side of Sophos, which has a mean and median completeness scores of 0.7 and 0.8 respectively). Interestingly, we find that those outliers are not the same outlier across all scanners, suggesting that an information-sharing paradigm, if implemented, would help improve the completeness score for those families. Finally, we notice that popular AV scanners, such as those widely used in the research community for evaluating the performance of machine learning based label techniques, provide mixed results: examples include VirusBuster, ClamAV, Symantec, Microsoft, and McAfee, which represent a wide range of detection scores. Note that those scanners are also major players in the AV ecosystem [28].

► **Completeness vs. diversity of labels.** Does the completeness as a score give a concrete and accurate insight into the performance of AV scanners? A simple answer to the question is negative. The measure, as defined earlier, tells how rich is an AV scanner with respect to the historical performance of the scanner but does not capture any meaning of accuracy. The accuracy of the AV scanners is determined by the type of labels assigned to each family, and whether those labels match the ground truth assigned by analysts upon manual inspection—which is captured by the correctness score. However, related to the completeness is the number of labels each AV scanner generates and the diversity (or perhaps the confusion) vector they add to the evaluation and use of AV scanners. For each AV vendor, we find the number of labels it assigns to each family. We then represent the number of labels over the various families as a boxplot (described above) and plot the results in Figure 3. The figure shows two interesting trends. First, while it is clear that no scanner with a non-empty detection set for the given family has a single label for all malware families detected by the scanner, the number of labels assigned by the scanner is always large. For example, the average number of labels assigned to a malware family by any scanner is 139, while the median number of labels is 69, which creates a great source of confusion. We further notice that one of the scanners (McAfee) had 2248 labels for the Avzhan malware family, which gives more than one label for every 2 samples. While we cannot statistically establish a confidence for the correlation of 0.24 between the number of labels and completeness—nor we can reject that as well— we observe some positive trend consistent for some of the scanners by visually comparing figures 3 and 2.

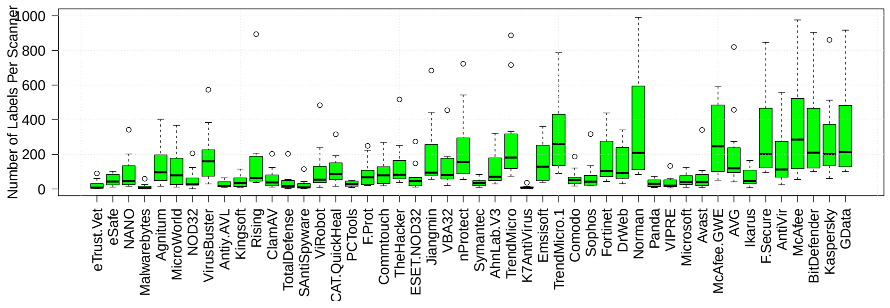

Fig. 3. A box plot of the number of labels assigned by the antivirus scanners used in the study for their detection of the malware families shown in Table 1. The y-axis is truncated (originally goes to 2248; smaller values are one indicator of better performance of an antivirus scanner.)

Completeness vs. largest label population size: Finally, for a deeper understanding of how the number of labels contributes to the completeness, we study the ratio of malware samples under the label with the largest population for every scanner. The results are shown in Figure 4. We see that while the average largest label among all we studied covers only 20% of the malware samples for any given scanner, some scanners, even with good completeness scores (e.g., Norman, Ikarus, and Avast, among others),

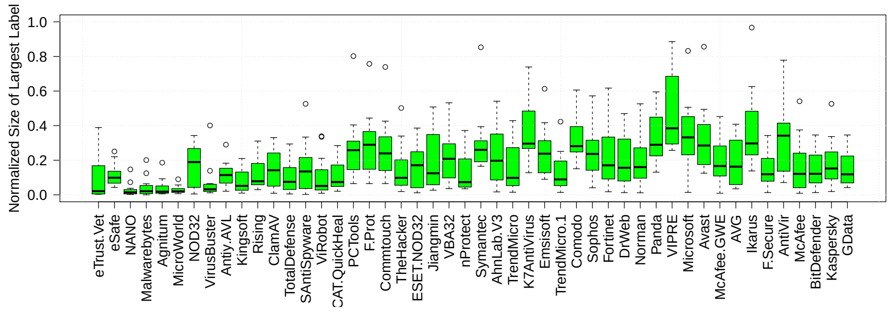

Fig. 4. A box plot of the size of the largest label of the given antivirus scanner for the various malware families shown in Table 1.

also provides a single label for the majority of detections (for 96.7% of the samples in Norman, for example). However, looking closer into the label given by the scanner, we find that it is too generic, and describes the behavior rather than the name known for the malware family; `Trojan.Win32.ServStart` vs `Avzhan`.

4.2 Correctness

Because of the large number of variables involved in the correctness, we limit our attention to two analysis aspects: general trends with a select AV vendor over all families, and then we demonstrate the correctness of two families for all vendors.

Family-based Trends We start the first part by iterating over each of the malware families, and group their behavior into three categories: families that AV scanners failed to label, labeled correctly, or labeled under other popular (unique) names.

► **Failed to label.** We observe that scanners totally mislabeled N0ise and Getkys. Among the generic labels of the first family, *krypt* and variants are used, where GData, BitDefend, and F-Secure provided coverage of 51.7%, 51.7%, and 50.8%, respectively. As for N0ise, Microsoft labeled it Pontoeb for 49% of the samples. We observe that Pontoeb shares the same functionality with N0ise. For both families, and in all of the labels provided by scanners, the most popular ones are too generic, including “trojan”, “virus”, “unclassified”, and nothing stands to correspond to functionality.

► **Labeled under known names.** Out of 3458 samples of Avzhan, the scanner AVG had the only meaningful label, which is `DDoS.ac`. Out of 3345 detections, 1331 were labeled with the meaningful label, corresponding to only about 39% of the samples. We notice that the rest of the AV scanners provide generic labels describing some of its behavior, like ServStart, which refers to the fact that the malware family is installed as a service. This lower result is observed despite the higher detection as observed in the AV scanners’ completeness performance on the family; an average of 71.5% and a median of 84.25%. We note that a generic label associated with the family, like *servicestart* (indicating the way of installation and operation of the sample) provides a collective correctness of label of about 62.7%, 47.5%, 46.6%, 41.8%, and 41.7% with Ikarus, Avast, NOD32, Emsisoft, and QuickHeal, respectively.

Each of Symantec, Microsoft, and PCTools detected Jkddos close to 98% of the time and labeled it correctly (as jackydos or jukbot, two popular names for the family) for 86.8%, 85.3%, and 80.3% of the time (Sophos followed with 42.3%). This correctness of labeling provides the highest performance among all families studied in this paper. The rest of the AV scanners labeled it either incorrectly or too generic, with the correct labels fewer than 5% of the time. As for DDoSer (also blackenergy), DrWeb provided close to 90% of detection, but only 64.1% of the total number of samples are labeled with the correct label, followed by 23.7% and 6.8% of correct labeling provided by Microsoft and Rising, and the rest of the scanners provided either incorrect or too generic labels like Trojan, generic, and autorun, among others.

ZeroAccess is labeled widely by the labels ZAccess, 0Acess, Sirefef, and Alureon, all of which are specific labels to the family. We find that while the detection rate of the family goes as high as 98%, the best correct labels are only 38.6% with Microsoft (other noteworthy scanners are Ikarus, Emsisoft, Kaspersky, and NOD32, with correctness ranging from 35.9% to 28.5%). Finally, Zeus is oftentimes labeled as Zbot, and we notice that while completeness score of 98% is obtained, only about 73.9% of the time the label is given correctly in a scanner (McAfee). Other well-performing scanners include Microsoft, Kaspersky, and AhnLab, providing correctness of 72.7%, 54.2%, and 53%, respectively.

► **Behavior-based labeling.** Lurid is labeled as Meciv, pucedoor, and Samkams by various scanners. Both of the first and second labels are for malware that drops its files on the system with names such as OfficeUpdate.exe and creates a service name like WmdmPmSp, while the last label is for worms with backdoor capabilities. This malware is labeled correctly based on the behavior, but not the name that is given to it originally in the industry. We notice that the top five scanners with the first and second labels are ESET-NOD32, Microsoft, Commtouch, F-port, and Rising, with correctness scores of 68.4%, 51.6%, 33.6%, 33.1%, and 31.1% respectively. When adding the third label, the top scanners include Symantec and PCTools, with 44.1% and 41.9%, respectively, at the third and fourth spots with the previous percent of top performing scanners unchanged, suggesting that the name samkams is specific to both scanners only.

DNSCalc is given two major labels, ldpinch and cosmu, covering about 34.2%, 34%, 33.7%, and 33.5% by Microsoft, TheHacker, Kaspersky, and ViRobot. However, both labels are generic and do not qualify for a correct label: ldpinch is a generic name for password stealing Trojans and cosmu is for Worm spreading capability.

The majority of AV scanners mislabel darkness as IRCBot (providing about 58.7% to 41.4% of correctness for the top five scanners). One potential reason to explain this mislabeling is that the source code of Darkness is public and shared among malware authors. Furthermore, as per the description above, the label is generic and captures a variety of worms based on the method of their propagation. Similarly, ShadyRAT is named as Hupigon by 10 scanners, with the highest AV scanner detecting it 70% of the time and giving it the correct label 30% of the time (43% of the detections).

Note that the type of the malware explains some of the differences in the correctness of labeling. For example, targeted and commercial malware families have lower correctness rates, potentially because AV vendors are less equipped to deal with them, and in some cases are less motivated to give them the proper labels since they are not seen as their main business. On the other hand, the mass-market malware (e.g., zeus) has better correctness score overall across multiple AV vendors (as shown in Figure 5).

AV-based Trends Now we turn our attention to showing the performance of every scanner we used over two selected malware families: Zeus and JKDDoS. We use the first family because it is popular, have been analyzed intensively, and is of particular interest to a wide spectrum of customers (e.g., banks, energy companies, etc). The second family is selected based on the performance highlighted in the previous section. The two families belong to financial opportunistic malware. To evaluate the correctness of the labels, we define three classes of labels: correct labels (based on the industrially popular name), generic labels (based on placeholders commonly used for labeling the family, such as "generic", "worm", "trojan", "start", and "'run"), and incomplete labels (including "suspicious", "malware", and "unclassified", which do not hold any meaning of a class). We plot the results of evaluating the scanners in Figure 5.

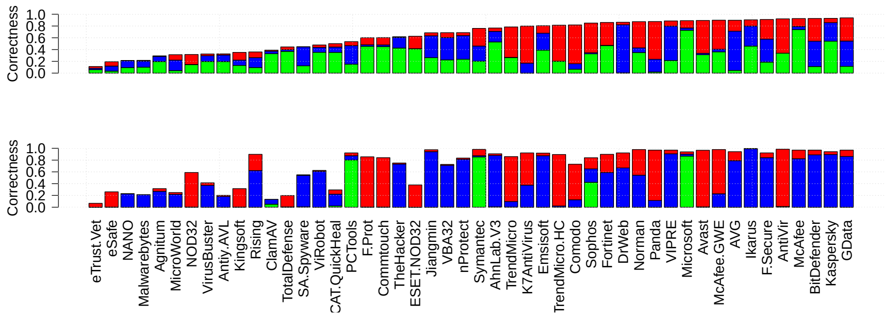

Fig. 5. Correctness score of all studied AV scanners— zeus (top) vs jkddos (bottom). The stacked bar plot legend is as follows: green for correct, blue for generic, and red for incomplete labeling. The score is computed out of the total number of samples (i.e., the maximum stacked bar length is equal to the completeness score of the given AV scanner for the studied family).

► **Zeus.** Although those labels are expected to give high scores—given their widespread—the results are mixed. In particular, each scanner labels a malware sample correctly 25.9% of the time on average. When considering generic names, the percent is increased to a total of 44.2%. When normalizing the correctness by the detections (rather than the number of samples, this yields a correctness score of 62.4%.

► **JKDDoS.** We notice that, while certain scanners perform well in detecting and giving the correct label for the majority of samples, as shown in the previous section, the majority of scanners mislabel the family. When considering the correct label, any scanner on average labels only 6.4% of the samples correctly. When adding generic labels, the percent is 45.1% on average (and 26.2% of mislabeled samples, on average), resulting in around 63% of correctness out of detections, and showing that the majority of labeled samples are either mislabeled or generically labeled.

This evaluation measure of AV scans has perhaps the most critical implication. In short, this measure says that, even when an AV provides a complete scan for a malware dataset, it is still not guaranteed that the same scanner will provide a correct result, and thus a labeling provided by an AV vendor cannot be used as a certain ground truth of labeling. On the other hand, findings in this section show that while on average the majority of scanners would perform poorly for a given malware family, it happens to be the case oftentimes that a few of them perform well by capturing the majority of samples in the studied sets. Those scanners vary based on the studied family, highlighting specialties by vendors with respect to malware families and labels, and suggesting that the added variety of scanners, while may help in increasing covering, only adds to the confusion under the lack of a baseline to guide their use.

4.3 Consistency

As defined in §2, the consistency score of an AV determines how it agrees with other scanners in its detection (or labeling; depending on metric used for inclusion of samples to a scanner) of malware samples. The consistency is determined per sample and is compared across all AV engines in a pairwise manner. This is, the Σ scanners we use in our study (48 in total) result in $\Sigma(\Sigma - 1)$ pairwise consistency scores in total, and $(\Sigma - 1)$ of them capture the consistency of each AV scanners with other scanners. We characterize those consistency scores by a box-plot that captures the first, second, and third quartiles, along with the maximum and minimum of the distribution of consistency score for the given AV scanner. In the following we highlight the findings concerning one family (Zeus) and using the detection (completeness) as the inclusion metric. We defer other combinations of options to the technical report, for the lack of space. The results are shown in Figure 6.

We observed (on average) that an AV engine is about 0.5 consistent with other AV engines, meaning that given a malware sample *detected* by $\mathcal{A}_i$, 50% of the time it is also detected by $\mathcal{A}_j$ as malicious. Figure 6 illustrates the consistency of each AV engine across all other engines using box plots (name of vendors are omitted for visibility). The figure clearly displays a median of approximately 50% for all AV engines. This finding further raises the question of how many AV scanners it would take to get a consistent detection for a given dataset, and the subtle problems one may face when utilizing multiple vendors for a given dataset.

Another observation we make is that there are 24 vendors consistent in their detection (almost perfectly) with a leading vendor in this particular family. There are several potential explanation for this behavior. It is likely that there is a mutual agreement of sharing, the 24 vendors scan the same set of samples as a single process, or perhaps that some of the vendors are following the lead of a single major vendor by populating hashes of malware. We emphasize that the observation cannot be generalized on all families, and when the phenomena is visible, the leading vendor changes.

4.4 Coverage

The coverage metric which we defined in §2 tells us how many AV vendors that we need to use in order to cover the largest number of samples possible in a dataset. The two versions we define for computing the coverage depend on the metric used for inclusion of samples to a given scanner: completeness and correctness.

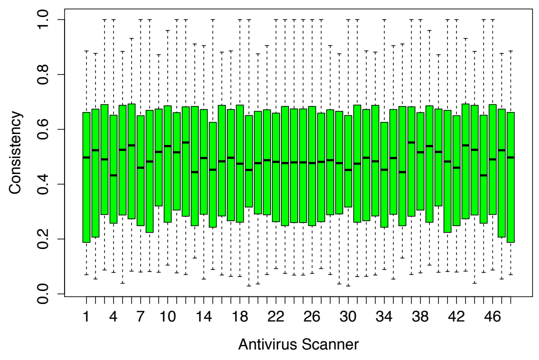

Fig. 6. Consistency of detections by 48 vendors (using the Zeus malware family).

► **How many scanners.** Again, we use the same vendors we used for plotting the previous figures of the completeness and correctness scores to answer this question. We use the approximation technique described in §2 to find the coverage, and highlight the findings by emphasizing the measurements for two families: Zeus and JKDDoS. Figure 7 shows the completeness and correctness-based coverage for two families. From

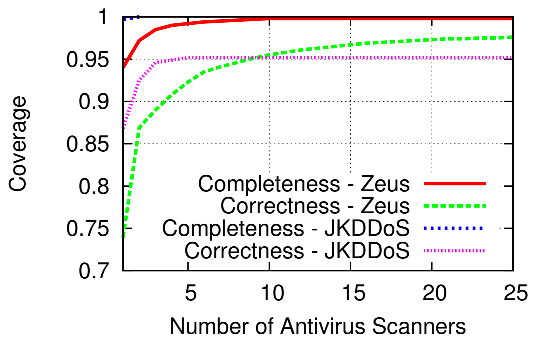

Fig. 7. The coverage using multiple AV scanners for Zeus and JKDDoS.

this figure, we make several observations. First, and as anticipated, we notice that the number of scanners we need to use in order to achieve a certain coverage score is higher for the correctness measure than the completeness. This finding is natural, and has been consistent with the relative order of the scores of individual scanners, since detecting a sample is not a guarantee for giving it the correct label, as we show in §4.1 and §4.2. Second, and more important, in both families we observe that a perfect (or close to perfect) completeness is not a guarantee for perfect correctness regardless of the number of AV scanners utilized for achieving the coverage. For example, while three vendors are enough for achieving a perfect completeness-based coverage for JKDDoS (and 10 are required in case of Zeus), the achieved correctness-based coverage in both cases using the same set of vendors is only 0.946 and 0.955. Even when all available vendors are used (48) together to cover the set of tested samples, a coverage of 0.952 and 0.976. This number does not change after using five and 25 vendors with JKDDoS and Zeus, respectively. Finally, we observe that this finding concerning the correctness-based coverage (regardless to the number of AV scanners we utilize) is consistent within a number of families, including browfox (shady RAT) and darkness.

5 Discussion

Findings in this paper call for further investigation on the implications on systems which use AV labels for their operation. Furthermore, those findings call for further investigations of how to make use of those labels, despite their shortcomings. In this section, we proceed to discuss the implications of the findings, ways to improve the labeling, and what we as a research community can do about those problems and directions. We set the suggestions as open research directions each of which deserve a separate study. We note that some of those directions are already touched upon in the past (academic and industry), although they were rarely adopted. We stress their benefits to the problem at hand and call the community to reconsider them with further investigation.

5.1 Implications

As mentioned in section 1, many systems rely in their operation on the labels produced by antivirus scanners for their operation. Those systems can be classified into two groups: 1) operational systems, and 2) academic proposals (e.g., systems to extrapolate labels of known malware samples to unlabeled ones). To this end, the implication of the findings in this study is two parts, depending on the targeted application.
• **Research applications:** for research applications that rely solely on AV labels for evaluating their performance, the findings in those paper are significant. Those systems, including the best known in the literature, use known and popular names of malware families in the industry. Accordingly, and based on the diversity of results produced by the various antivirus scanners used in the literature for naming malware samples, one would expect the accuracy of those systems not to hold as high as claimed.
• **Security operations:** As for the operation systems that rely on labels produced by antivirus scanners, the findings in this paper are *warning* and *call for caution* when using those labels for decision-making. We note that, however, security analysts in typical enterprises know beyond what academic researchers know of malware families, and can perhaps put countermeasures into action by knowing the broad class of a malware family, which is oftentimes indicated by the labels produced by antivirus scanners. Notice that this is not a knowledge gap, but rather a gap in objectives between the two parties. Furthermore, operational security analysts oftentimes employ conservative measures when it comes to security breaches, and knowing only that a piece of code is "malicious" could be enough to put proactive countermeasures into actions. However, we

emphasize that even the approximate names and generic classes of labels take time to get populated in antivirus scans, which in itself may have an effect on operational security. Finally, another class of operational security, namely the law enforcement efforts which rely on names for online crime attribution, maybe impacted by the limitations of AV labels highlighted in this work.

5.2 Remedies

Efforts to improve the labeling and the way they are used for serving the security of individual and enterprises can be split into two directions: research and industry. In the following, we stress several remedies and effort that can address the problem. Notice that some of those directions are previously suggested, however they are not widely adopted for various reasons, including the lack of interest and incentives. To that end, we compile the list of remedies to stress their relevance to the problem at hand, and that the research community can further contribute by pursuing those directions.

• **Data sharing:** most studies for classifying or clustering malware samples into specific families require a solid ground truth. In reality, and if any of those systems to be realized operationally, the ground truth is not needed for the entire studied or analyzed data, but rather for at least a portion of it to 1) establish a baseline of accuracy, and 2) to help tune those systems by exploring discriminative features to tell malware families apart. Despite the drawbacks of benchmarking, a step that might help remedy the issues raised in this study is by sharing data with such solid ground truth to evaluate those academic systems on it. Despite some recent initiatives in enabling data sharing, transparency with respect to that is still one of the main challenges that face our community and platforms has to be explored for enabling and facilitating such efforts.

• **Names unification:** many of the names provided by antivirus scanners are inaccurate as a side effect of the techniques used for creating them. For example, static signatures that are fully automated give a generic name that often does not capture a specific family. The same signature often results in different names, based on the vendor. One way to help increasing the consistency and accuracy of names is to create such a naming convention that can followed by multiple players in the antivirus ecosystem.

• **Making sense of existing names:** names given to malware families sometimes exhibit the lack of a standard convention of naming. Having a convention, while help addressing the problem in the future, may not address it for already labeled samples. To this end, the research community can help by making sense of various names given to malware samples by various vendors. Techniques with potential of resolving naming conflicts include voting, vendor reputation, and vendor accuracy and influence for a specific family, and other techniques such as those utilized by VAMO [31].

• **Indicators sharing:** while there are multiple forms and platforms for sharing threat indicators that can be used for accurately naming malware families and classes, those indicators are less used in the community. Enabling the use of those sharing platforms to realize intelligence sharing can greatly help accurately and actively name malware families with less chances of name conflict.

• **What is a name?** Rather than a generation of the family or a historical background-driven name that has little chances of adoption by variety of vendors, perhaps it is more important to give a broad, but meaningful, name of a class for the malware family.

Those names can be driven based on the functionality and purpose of the malicious code, rather than the background story of family as it is the case of many of the names used with malware families (including those analyzed in the paper).

6 Related Work

AV labels have been widely employed in the literature for training algorithms and techniques of malware classification and analysis [6, 7, 15, 18, 25, 29, 30, 32, 33, 37, 39, 42] (a nice survey of many of those works is in [34]). However, there is less work done on understanding the nature of those labels. To the best of our knowledge, the only prior work dedicated for systematically understanding AV-provided labels is due to Bailey et al. [6]. However, our work is different from that work in several aspects highlighted as follows. First, while our work relies on a set of manually-vetted malware samples with accurate label and family association, the work in [6] relies on an AV vendor as a reference. Second, our study considers the largest set of AV-vendors studied in the literature thus far for a comparative work. Finally, given that we rely on a solid ground truth, we develop several metrics of AV scans evaluation that are specific to our study that are not considered before.

Related to our work is the work of Canto et al. [8], which tries to answer how difficult it is to create a reference and representative data set of malware. The authors suggest that while one can create a dataset that is representative at a certain time, there is no guarantee that the same dataset would be representative in the future. The work also highlights labeling inconsistency on a limited set of samples over two vendors. Our work, on the other hand, quantifies the inconsistency in labeling against a reference dataset. VAMO [31] is a yet another related work in addressing shortcomings of malware labeling for research validation, and in introducing that tries to make sense of AV labels. VAMO introduces a method that constructs a graph from the labels provided by AV vendors, define a distance between labels, and group those that are close in distance into the same label. An issue that VAMO overlooks is that it still relies on those labels provided by AV vendors as a ground truth for grouping malware samples. Unlike the work of Canto et al. [8], for example, which highlights inconsistencies in labeling against a fixed sample label, VAMO does not consider a reference label for evaluating how good is their grouping.

7 Conclusion and Future Work

In this work, we unveil the danger of relying on incomplete, inconsistent, and incorrect malware labels provided by AV vendors for operational security and in the research community, where they are used for various applications. Our study shows that one needs many independent AV scanners to obtain complete and correct labels, where it is sometimes impossible to achieve such goal using multiple scanners. Despite several limitations (in §1), our study is the first to address the problem and opens many future directions. An interesting by-product of our study is several recommendations and open directions for how to answer the shortcomings of today's AV labeling systems. In the future, we will look at methods that realize this research and answer those directions by tolerating across-vendors inconsistencies, and overcome the inherit incompleteness and incorrectness in labels. We hope this work will trigger further investigation and attention in the community to this crucial problem.

Acknowledgement. The work of the second author was done while he was with VeriSign Inc. We would like to thank Matt Larson and Danny McPherson for their involvement in an earlier stage of this research, Burt Kaliski and Allison Mankin for their feedback on an earlier version, the iDefense team for providing the data, and our shepherd, Jose M. Fernandez, for his insightful remarks on improving this work.

References

1. —. ZeroAccess. `http://bit.ly/IPxi0N`, July 2011.
2. —. Sykipot is back. `http://www.alienvault.com/open-threat-exchange/blog/sykipot-is-back`, July 2012.
3. Arbor Networks. Another family of DDoS bots: Avzhan. `http://bit.ly/IJ7yCz`, September 2010.
4. Arbor Networks. JKDDOS: DDoS bot with an interest in the mining industry? `http://bit.ly/18juHoS`, March 2011.
5. Arbor Networks. A ddos family affair: Dirt jumper bot family continues to evolve. `http://bit.ly/JgBI12`, July 2012.
6. M. Bailey, J. Oberheide, J. Andersen, Z. Mao, F. Jahanian, and J. Nazario. Automated classification and analysis of internet malware. In *RAID*, 2007.
7. U. Bayer, P. M. Comparetti, C. Hlauschek, C. Krügel, and E. Kirda. Scalable, behavior-based malware clustering. In *NDSS*, 2009.
8. J. Canto, M. Dacier, E. Kirda, and C. Leita. Large scale malware collection: lessons learned. In *IEEE SRDS Workshop on Sharing Field Data and Experiment Measurements on Resilience of Distributed Computing Systems*, 2008.
9. Damballa. The IMDDOS Botnet: Discovery and Analysis. `http://bit.ly/1dRi2yi`, March 2010.
10. DDoSpedia. Darkness (Optima). `http://bit.ly/1eR40Jc`, December 2013.
11. I. Gashi, V. Stankovic, C. Leita, and O. Thonnard. An experimental study of diversity with off-the-shelf antivirus engines. In *Network Computing and Applications, 2009. NCA 2009. Eighth IEEE International Symposium on*, pages 4–11. IEEE, 2009.
12. Jose Nazario. BlackEnergy DDoS Bot Analysis. `http://bit.ly/1bidVYB`, October 2007.
13. Kelly Jackson Higgins. Dropbox, WordPress Used As Cloud Cover In New APT Attacks. `http://ubm.io/1cYMOQS`, July 2013.
14. D. Kerr. Ubisoft hacked; users' e-mails and passwords exposed. `http://cnet.co/14ONGDi`, July 2013.
15. J. Kinable and O. Kostakis. Malware classification based on call graph clustering. *Journal in computer virology*, 7(4):233–245, 2011.
16. D. Kong and G. Yan. Discriminant malware distance learning on structural information for automated malware classification,. In *Proceedings of the 19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2013.
17. P. Kruss. Complete zeus source code has been leaked to the masses. `http://www.csis.dk/en/csis/blog/3229`, March 2011.

18. A. Lanzi, M. I. Sharif, and W. Lee. K-tracer: A system for extracting kernel malware behavior. In *NDSS*, 2009.
19. F. L. Lévesque, J. Nsiempba, J. M. Fernandez, S. Chiasson, and A. Somayaji. A clinical study of risk factors related to malware infections. In *ACM Conference on Computer and Communications Security*, pages 97–108, 2013.
20. F. Maggi, A. Bellini, G. Salvaneschi, and S. Zanero. Finding non-trivial malware naming inconsistencies. In *Information Systems Security*, pages 144–159. Springer, 2011.
21. Malware Intel. n0ise Bot. Crimeware particular purpose for DDoS attacks. `http://bit.ly/1kd24Mg`, June 2010.
22. mcafee.com. Revealed: Operation Shady RAT. `http://bit.ly/IJ9fQG`, March 2011.
23. Microsoft - Malware Protection Center. Spyeye. `http://bit.ly/1kBBnky`, Dec 2013.
24. A. Mohaisen and O. Alrawi. Amal: High-fidelity, behavior-based automated malware analysis and classification. Technical report, VeriSign Labs, 2013.
25. A. Mohaisen and O. Alrawi. Unveiling zeus: automated classification of malware samples. In *WWW (Companion Volume)*, pages 829–832, 2013.
26. NYTimes. Nissan is latest company to get hacked. `nyti.ms/Jm52zb`, Apr. 2013.
27. J. Oberheide, E. Cooke, and F. Jahanian. Cloudav: N-version antivirus in the network cloud. In *USENIX Security Symposium*, pages 91–106, 2008.
28. OPSWAT. Antivirus market analysis. `http://bit.ly/1cCr9zE`, December 2012.
29. Y. Park, D. Reeves, V. Mulukutla, and B. Sundaravel. Fast malware classification by automated behavioral graph matching. In *CSIIR Workshop*. ACM, 2010.
30. R. Perdisci, W. Lee, and N. Feamster. Behavioral clustering of http-based malware and signature generation using malicious network traces. In *USENIX NSDI*, 2010.
31. R. Perdisci and M. U. Vamo: towards a fully automated malware clustering validity analysis. In *ACSAC*, pages 329–338. ACM, 2012.
32. K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov. Learning and classification of malware behavior. In *DIMVA*, pages 108–125, 2008.
33. K. Rieck, P. Trinius, C. Willems, and T. Holz. Automatic analysis of malware behavior using machine learning. *Journal of Computer Security*, 19(4):639–668, 2011.
34. C. Rossow, C. J. Dietrich, C. Grier, C. Kreibich, V. Paxson, N. Pohlmann, H. Bos, and M. van Steen. Prudent practices for designing malware experiments: Status quo and outlook. In *IEEE Sec. and Privacy*, 2012.
35. M. I. Sharif, A. Lanzi, J. T. Giffin, and W. Lee. Automatic reverse engineering of malware emulators. In *IEEE Sec. and Privacy*, 2009.
36. V. Silveira. An update on linkedin member passwords compromised. `http://linkd.in/Ni5aTg`, July 2012.
37. W. T. Strayer, D. E. Lapsley, R. Walsh, and C. Livadas. Botnet detection based on network behavior. In *Botnet Detection*, 2008.
38. Symantec. Advanced persistent threats. `http://bit.ly/1bXXdj9`, December 2013.
39. R. Tian, L. Batten, and S. Versteeg. Function length as a tool for malware classification. In *IEEE MALWARE*, 2008.
40. Trend Micro. Trend Micro Exposes LURID APT. `http://bit.ly/18mX82e`, September 2011.
41. A. G. West and A. Mohaisen. Metadata-driven threat classification of network endpoints appearing in malware. In *DIMVA*, 2014.
42. H. Zhao, M. Xu, N. Zheng, J. Yao, and Q. Ho. Malicious executables classification based on behavioral factor analysis. In *IC4E*, 2010.