AV-Meter: An Evaluation of Antivirus Scans and Labels

Aziz Mohaisen[1] and Omar Alrawi[2⋆]

[1]Verisign Labs [2]Qatar Foundation

Abstract. Malware detection and labeling are two major goals of antivirus (AV) vendors. AV scanners are designed to detect malware and to label their detection 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, researches relied for so long on AV labels to establish a baseline of ground truth to compare their detection, classification, and clustering algorithms, despite many papers pointing out the subtle problem of relying on AV labels. Ironically, the literature lacks any prior systematic work on validating the performance of antivirus vendors, and the reliability of those labels (or even detections).
In this paper, we set out to answer several questions concerning the completeness, correctness, and consistency of AV detections and labels. 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 negative (and perhaps scary) results, and invite the community to challenge assumptions about relying on antivirus scans and labels as a ground truth for malware analysis and classification. We also suggest several directions and open research directions to help addressing the problem.

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

1 Introduction

Antivirus (AV) companies continuously evolve to improve their products which provide users with an added protection from malicious software (malware) threats. However, AV products are not a complete solution: malware evolves at a much faster rate than AV products, which then forces AV companies to innovate and use smarter approaches for malware detection [42]. AV products provide two major functionalities: *detection* and *labeling* [25]. AV scanners are designed to detect malware and to label the detection based on a family association [6]. Labeling is an important feature to AV vendors, with many applications [11]. For example, labeling allows AV vendors to filter known malware and focus on new or variants of familiar malware families with known remedies. Labeling also enables the AV vendors to track a malware family and its evolution, thus allowing them to proactively create and deploy disinfection mechanisms of emerging threats [23]. Also, in security operations practitioners have an interest in identifying a

⋆ This work was done while the author was with Verisign.

malware by a family name so that they can mitigate the threat for their organization. Last but not least, researchers have benefited from detections and labeling of malware provided by AV vendors in many ways. For instance, researchers in the fields of malware analysis, detection, and classification have benefited from AV scans and labels in establishing baselines to compare their designs against [6, 7, 16, 28, 29, 32, 37, 40, 41] (a survey is in [31]).

1.1 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 [26]. Despite various benefits [10], 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 as they are discovered [19]. Malware names are usually created by analysts who study new malware samples, by utilizing artifacts within the malware to derive and give them names. Some malware families are so popular in underground forums, like SpyEye [22], Zeus [17], ZeroAccess [1], DirtJumper [5], etc., that AV vendors use those names given to the malware by their authors or the underground market. Other smaller and less prominent malware families are usually named independently by each AV company. For example, targeted malware, which is known as advanced persistent threat (APT) [36], is low key that AV vendors track independently, usually resulting in different naming.

The diversity of the market with the multistakeholder 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 is initially detected using a static signature, then later due to its polymorphism technique it is heuristically using a generic malicious behavior. In such case, the AV vendor will give it another label creating an inconsistent label within the same AV vendor's labeling schema. These inconsistencies and shortcomings may not have a direct implication on the malware detection provided by the AV scanner, although they impact applications that use AV labeling.

1.2 Inconsistencies Create Inefficiencies

For example, the use of AV labels for validating malware classification research—while creating a ground for comparing different works to each other—has many shortcomings and pitfalls. Malware samples collected by researchers are oftentimes not necessarily 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 used by those engines. 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 [23], making such translation sometimes impossible.

The detection and labeling inconsistencies create inefficiencies in the industry that could prevent stakeholders from benefitting from a common standard for malware family naming and information sharing. 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 millions of dollars in intellectual property theft, direct and indirect costs, or reputation damage. While companies are conservative in revealing such information about the compromise of their systems and exfiltration of their users' or proprietary data, and only insiders are aware of this threat and its cost, there has been recent public information that support and highlight the trend. Examples of such incidents include the hacking of LinkedIn [34], Ubisoft [14], LivingSocial [33], and most famously Nissan [24].

1.3 An "Elephant in the Room"

Sadly, while we are not the first to observe those inefficiencies in AV labeling systems [6, 7, 31]. the community so far spent so little time systematically understanding them, let alone quantifying the inefficiencies and providing solutions to address them. Even more ironic, some of those works that pointed out the problem with AV labels used the same labels for validating algorithms by establishing a baseline and a ground truth to compare their work to [7, 31]. 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 (§3.1). We obtain those samples from operations in a large security firm (§3.2), where vetting and highly accurate techniques for malware family labeling are employed as part of a paid 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, but not testing the correctness of those labels or the completeness of the detections provided by different engines. Accordingly, we develop metrics to evaluate the completeness, correctness, consistency, and coverage (defined in §2), and use them to evaluate the performance of various vendors. Our measurement study does not trigger active scans, but rather depends on querying the historical detections provided by each AV engine. We show that, while AV scanners are intended mainly for detection, and are supposed to provide a perfect detection, many of them provide poor completeness results, indicating less than perfect detection. We show those findings beyond doubts, by demonstrating that any sample we test exists in at least one AV scanner, thus one can obtain full coverage of the tested samples using multiple vendors.

1.4 Contribution and Limitations

To this end, the contribution of this study is twofold. We provide metrics for evaluating AV detections and labeling systems. Second, we use a highly-accurate and manually-vetted dataset for evaluating the detections and labelings of large number of AV engines

using the proposed metrics. The dataset, scripts, and AV scans will be all made available publicly to the community to use and contribute to problem at hand. 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.

Our study has many limitations to it, 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 don't use every hash in every given AV scanner. Similarly, the same generalization cannot be used for the malware families, since we didn't use all samples known by the AV scanners. Our study is, however, meaningful in answering the limited context's questions it poses for 12000 malware samples that belong to various timely families. Furthermore, our study goes beyond the best known work in the literature in the problem by not relying on AV-provided vendors as reference for comparing other vendors (further details are in §6).

1.5 Organization

The organization of the rest of this paper is as follows. In section 2 we suggest 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: we first introduce evaluation metrics for AV vendors, and then use those metrics to evaluate 48 vendors and their performance on our dataset. In section 5 we discuss implications of the findings and remedies. In section 6 we review the related work, followed by concluding remarks, open directions, 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 $\mathcal{S}''_i$). 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 of an AV vendor as the number detections returned by the vendor 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 vendor as the number of detections returned by the vendor 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 vendors 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 vendors 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, and by ignoring edge cases, 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, for the completeness and correctness respectively, we have:

$$\min_k \left\{ \bigcup_{a_j \in \mathcal{A}_k} \mathcal{S}'_{ij} = \mathcal{D}_i \right\} \text{ and } \min_k \left\{ \bigcup_{a_j \in \mathcal{A}_k} \mathcal{S}''_{ij} = \mathcal{D}_i \right\}$$

The problem is well known in the literature, and is called the minimal set cover problem known to be NP-hard [39]. A factor-$\log \Delta_i$ approximation algorithm to the problem exists, and works by iteratively selecting the subset in $\mathcal{S}'_i$ (or $\mathcal{S}''_i$ for correctness) that has the maximal overlap with the current set of uncovered elements in $\mathcal{D}_i$. This is, we start with the set of the largest overlap with $\mathcal{D}_i$ (namely, $\mathcal{S}'_{ij}$), add it to the set cover candidates of results, and omit its elements from $\mathcal{D}_i$. For $\mathcal{D}_i \setminus \mathcal{S}'_i$, we find the subset with the largest overlap with it, and add it to the results. We repeat by selecting subsets in $\mathcal{S}_i$ until the set $\mathcal{D}_i$ is totally covered (i.e., the remainder is an empty set)—upon which the cover size is the number of subsets eliminated—or the subsets in $\mathcal{S}_i$ are totally exhausted.

Related to the both completeness and correctness score we computed above 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 the all labels provided by an AV scanner for each malware family. We compute those derived metrics for each AV 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 12 distinct malware families[2]. 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.

In the following, we elaborate on each of those families.

- **Zeus:** Zeus is a banking Trojan that targets financial sector by stealing credentials from infected victims. The malware steals credentials by hooking Windows API fucntions 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 [8], 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:** also known as Optima, is a malware family that is availably commercially and is 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 [9].
- **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:** n0ise is a DDoS tool with extra functionalities like stealing credentials from victim 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 [20].
- **ShadyRat:** is a targeted malware that is used to steal sensitive information like trade secrets, patent technologies, and internal documents. The malware employes a stealthy technique when communicating with the C2 by using a combination of encrypted HTML comments in compromised pages or stegonagraphy in images uploaded to a website [21]

[2] We intend to make the dataset public.

– **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 group is also known as APT12 by Mandiant. The malware steals sensitive information and targets research sector [13].
– **Lurid:** was first observed by the Japanese software security vendor Trend Micro on September 2011. 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 indicator that perhaps the family is being used commercial as a hit man [38]
– **Getkys:** (also known as 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]

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

3.2 Samples Vetting and Labeling

Each malware sample in each of those samples has been identified manually by analysts over a period of time in a service that requires reverse engineering and manual vetting. Our dataset consists of variety of families and a large number of total samples, which enable 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[3]. To identify the label of this family, we used forensic memory signatures to identify a set of possible samples that belong to the given family from our malware repositories, then we manually vetted the set to ensure our final data set is clean of malware families that might falsely trigger our memory signatures. 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. We give each family we use in this study the name most popular and accepted in the industry based on a domain-knowledge model.

3.3 VirusTotal

VirusTotal is a multi-engine AV scanner that accepts submissions by users and scans the sample with multiple AV 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 note that because no researchers had an alternative to what the AV scanners provide, so far the completeness, consistency, and correctness—the three comparison and evaluation measures we study in §4—of AV labels and scans were not challenged, and implications of those metrics of an AV scan were overlooked in the past.

4 Measurements and Findings

In the following, we present the results and provide some remarks on them.

4.1 Completeness

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.

Overall completeness scores: Figure 1 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

[3] 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).

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), and is never perfectly complete, indicating that even the best and most populated scanner may miss some of the samples studied in this work. 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). In other words, every scanner on average misses 40% of the studied malware families, and cannot be used as a single source for determining whether a set of malware samples is benign or malicious in total. While researchers strive to achieve 99% of accuracy in their classification of malware samples [7], a 40% of incompleteness is an overlooked margin of error, and it is unclear how this margin is considered in the total performance measures in the literature. Even worse, the completeness of a scan does not guarantee a correctness of a detection.

Furthermore, the same figure shows that even with the well performing scanners on the majority of samples and families, there are always families that are missed by the majority of scanners, 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—while generally are targeted malware that take longer to propagate in AV scanners—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 results across the board and are by no means consistently better than other scanners: examples include VirusBuster, ClamAV, Symantec, Microsoft, and McAfee, which represent a wide range of detection scores.

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)

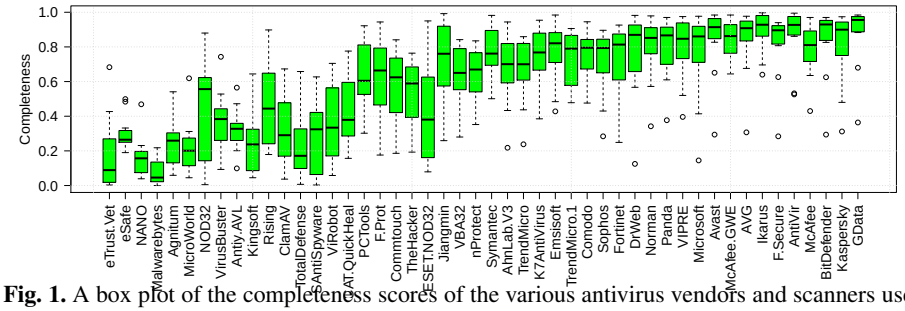


Fig. 1. A box plot of the completeness scores of the various antivirus vendors and scanners used in the study against the 11 malware families shown in Table 1. The y-axis is on the linear scale, of 0-1.

and plot the results in Figure 2. 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 are oftentimes 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. The second trend we see on the same figure is that the number of labels assigned by the scanner is positively correlated with the completeness score (by visually comparing figures 2 and 1; correlation coefficient of 0.24).

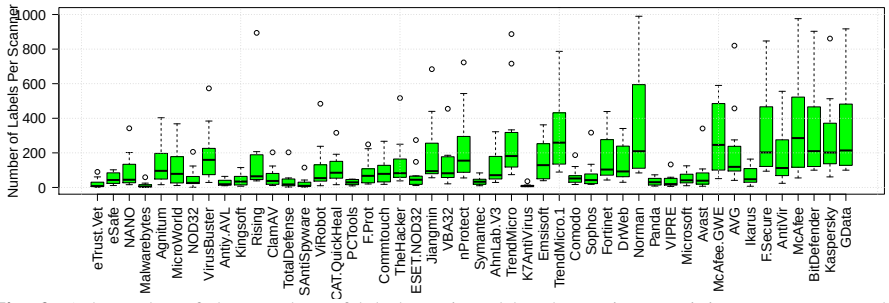


Fig. 2. A box plot of the number of labels assigned by the various antivirus scanners used in the study for their detection of the malware families shown in Table 1. The y-axis is truncated (originally goes to 2248; smaller values are one indicator of better performance of an antivirus scanner.)

Completeness vs. largest label: Finally, for a deeper understanding of how the number of labels contribute to the completeness (and later accuracy of labeling), we study the ratio of malware samples associated with the largest label given by each scanner. The results are shown in Figure 3. 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,

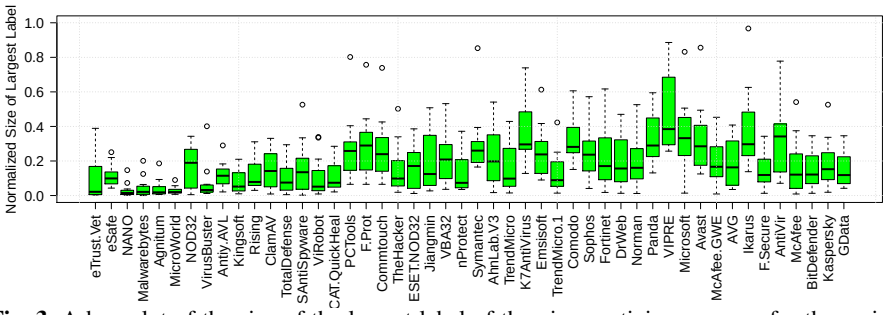


Fig. 3. A box plot of the size of the largest label of the given antivirus scanner for the various malware families shown in Table 1.

even with good completeness scores (e.g., Norman, Ikarus, and Avast, among others), 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

We define the correctness of the labeling provided by an AV scanner as ratio of correctly labeled malware samples out of the total number of samples in the studied family (with respect to the reference label). 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, 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: First, we notice that two of the families studied in this paper had no correct labels in any of the scanners: N0ise and Getkys. Of the generic labels associated with the first family is *krypt* and variants (a generic label corresponding to obfuscated malware samples), with GData, BitDefend, and F-Secure providing coverage of the label of 51.7%, 51.7%, and 50.8%, respectively. With the alternative but unique names associated with N0ise, Microsoft labels it correctly as Pontoeb 49% of the time (out of the studied malware samples). We observe that Pontoeb shares the same functionality with N0ise. In all of the incorrect labels provided by scanners, the most popular ones are too generic, including “trojan”, “virus”, “unclassified”, and nothing stands to correspond to a functionality or behavior.

Labeled under known names: Second, 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 poor result is observed despite the reasonable 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 best 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 under 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.

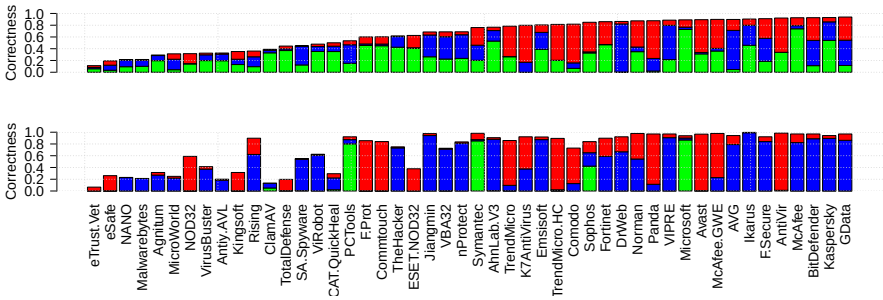


Fig. 4. Correctness score of all studied AV scanners— zeus (top) vs jkddos (bottom). The stacked bar plot legend is as follows: green for correct, blue for generic, and red for incorrect labeling. The score is computed out of the total number of samples (i.e., the maximum stacked bar length is equal to the completeness score of the given AV scanner for the studied family).

Behavior-based labeling: Third, Lurid is labeled as *Meciv*, *pucedoor*, and *Samkams* by various scanners which are based on the behavior of the malware. Both of the first labels are for malware that drops its files on the system with names such as OfficeUpdate.exe and creating 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 performing scanners when considering 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 as a correct name of the sample, the list of top performing scanners includes Symantec and PCTools, with 44.1% and 41.9%, respectively, at the third and fourth spots with the previous percents 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 scanners.. 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 a generic label for Worm spreading capability.

Darkness is mislabeled as IRCBot (a detection for worms that spread using the Internet Relay Chat–IRC), by the majority of the scanners (providing about 58.7% to 41.4% of correctness for the top five scanners). One potential reason to explain this mislabeling is the fact that the source code of Darkness is public and shared among malware authors (thus, signatures applied to it confuses it with other families). 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).

AV-based Trends Now we turn our attention to understanding and demonstrating 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 who analyzed and understand the family well. The second family is selected based on the performance of top scanners 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 highlight in the previous section), generic labels (based on placeholders commonly used for labeling the family, such as “generic”, “worm”, “trojan”, “start”, and “’run”, which partly indicate mechanisms of operation), and bad labels (including “suspicious”, “malware”, and “unclassified”, which do not hold any meaning of a class). We plot the results of evaluating the vendors in Figure 4.

For Zeus, although those labels are expected to yield high results—given that the family is well understood and is of high-interest—the results show otherwise. In particular, we find that on average, each scanner labels a malware sample correctly 25.9% of the time. Furthermore, only 18.3% of increased correctness is added by considering generic names, brining up the correctness to 44.2% (and missing 20.6 of the labels). When normalizing the correctness by the detections (rather than the total number of the samples), this yields a correctness score of only 62.4%.

JKDDoS does not seem to bring any positive insight into the performance of AV scanners. 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 perform poorly on the sample. When considering the correct label, on average only 6.4% of the samples are labeled correctly by any scanner. When adding generic labels, the percent increases to 45.1% on average (and 26.2% of mislabeled samples, on average) maintaing 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 specialities 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 $(\Sigma - 1)^2$ 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 5.

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 5 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 is always a set of vendors who are always consistent with a major vendor. For example, we observe that 24 vendors are consistent in their detection (almost perfectly) with the 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 same set of samples is scanned by the 24 vendors as a single

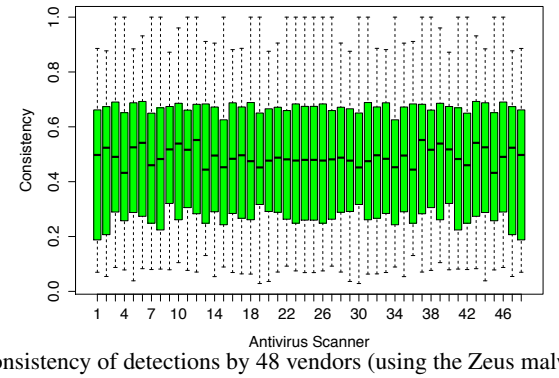


Fig. 5. Consistency of detections by 48 vendors (using the Zeus malware family).

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.

How many AV 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 [39] described in §2 to find the coverage of the various malware families. We review the results of all families, and emphasize the measurements for two families: Zeus and JKDDoS.

Figure 6 shows the completeness and correctness-based coverage for two families. From 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 imperfect

correctness-based coverage (regardless to the number of AV scanners we utilize) is consistent in a number of malware families, including browfox (shady RAT), darkness, and others.

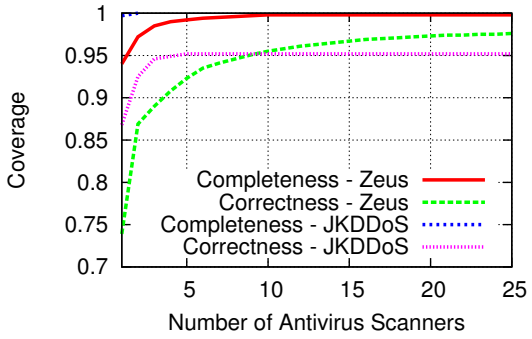


Fig. 6. The coverage when using multiple AV vendors on two families: Zeus and JKDDoS. The coverage is computed using the approximation mechanism described earlier, for the correctness and completeness.

5 Discussion

So far, findings presented in this paper focused on the negative aspects of the performance of AV vendors, and they indicate that the labels produced by AV scanners to name malware samples are incomplete, inconsistent, and oftentimes incorrect. These findings, however, call for further investigation on the implications on systems which use those 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 about those problems and directions. We set the suggestions as open research directions each of which deserve a separate study.

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 claim accuracy of 99% of classification and clustering of malware samples into specific families, the findings in those paper are *critical*. Those systems, including the best known in the literature, use known and popular names of malware families in the industry (including those named in this paper). 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 in those studies.

• **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. 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. Furthermore, we note that even with the most popular malware family among the ones we studied in this work, the completeness score is oftentimes less than perfect, and that in itself calls for further caution.

5.2 Remedies

The next question that emerges as we discuss the implications of the study is what we, as a community of researchers and industry, can do about the findings in this work, and the question we raised so far. As with the implications, 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 outline several remedies and effort that can address the problem if taken by the intended parties.

• **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 initives 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:** while many of the names provided by antivirus scanners are inaccurate due to the lack of knowledge of studied malware samples—e.g., the static signature used analyzing scanning the sample to give it a name is too generic and doesn't capture a specific family—an equally important and contributing factor to the confusion in this domain is the diversity of (possibly valid) names given by competitors malware families as they discover them. 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:** also related to the previous direction, we note that oftentimes names given to malware families are the result of the lack of a standard convention of naming. Having this convention in the future will help name malware samples but will not fix the mess of names of already analyzed large libraries of malware samples. To this end, the research community can help by making sense of var-

ious names given to malware samples by various vendors to create such convention. This would be enabled if highly accurate malware labels available in various institutes (including those studied in this paper) are shared to the community interested in analyzing them. Techniques with potential of resolving naming conflicts by various vendors include voting, vendor reputation and scoring, and evolution of vendor accuracy and influence for a given family.

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

Ironically, while the use of AV-provided labels has been widely employed in the literature for training algorithms and techniques utilized for malware classification and analysis [6, 7, 15, 18, 23, 27–30, 35, 37, 41] (a nice survey of many of those works is in [31]); techniques that are intended for accurately labeling malware samples, there is less work done on understanding the nature of those labels, while less is done in this direction by only pointing out issues with AV-provided 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:

– While our work relies on a set of manually-vetted malware samples for which we know the accurate label and family, the work in [6] relies on an AV vendor as a reference. In particular, the authors use McAfee as the (complete and accurate) reference of detection and labeling and compare other vendors to it. Our technique avoids this issue by relying on a manually inspected reference set.
– Our study considers the largest set of AV-vendors studied in the literature thus far for a comparative work. We do that by relying on the largest number of manually-vetted malware samples as well. As shown in the study, even when certain AV providers are consistent among each other, they still don't provide perfect results with respect to the ideal ground truth.
– 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..

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 §1.4), 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 will make public all datasets and codes used in this study to help pursue alternatives. We hope this work will trigger further investigation and attention in the community to this crucial problem.

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. Damballa. The IMDDOS Botnet: Discovery and Analysis. http://bit.ly/1dRi2yi, March 2010.
9. DDoSpedia. Darkness (Optima). http://bit.ly/1eR40Jc, December 2013.
10. 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.
11. M. Howard and S. Lipner. *The security development lifecycle*, volume 11. Microsoft Press, 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. Maggi, A. Bellini, G. Salvaneschi, and S. Zanero. Finding non-trivial malware naming inconsistencies. In *Information Systems Security*, pages 144–159. Springer, 2011.
20. Malware Intel. n0ise Bot. Crimeware particular purpose for DDoS attacks. `http://bit.ly/1kd24Mg`, June 2010.
21. mcafee.com. Revealed: Operation Shady RAT. `http://bit.ly/IJ9fQG`, March 2011.
22. Microsoft - Malware Protection Center. Spyeye. `http://bit.ly/1kBBnky`, December 2013.
23. A. Mohaisen and O. Alrawi. Unveiling zeus: automated classification of malware samples. In *WWW (Companion Volume)*, pages 829–832, 2013.
24. New York Times. Nissan is latest company to get hacked. `http://nyti.ms/Jm52zb`, April 2013.
25. J. Oberheide, E. Cooke, and F. Jahanian. Cloudav: N-version antivirus in the network cloud. In *USENIX Security Symposium*, pages 91–106, 2008.
26. OPSWAT. Antivirus market analysis. `http://bit.ly/1cCr9zE`, December 2012.
27. Y. Park, D. Reeves, V. Mulukutla, and B. Sundaravel. Fast malware classification by automated behavioral graph matching. In *CSIIR Workshop*. ACM, 2010.
28. R. Perdisci, W. Lee, and N. Feamster. Behavioral clustering of http-based malware and signature generation using malicious network traces. In *USENIX NSDI*, 2010.
29. K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov. Learning and classification of malware behavior. In *DIMVA*, pages 108–125, 2008.
30. 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.
31. 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.
32. M. I. Sharif, A. Lanzi, J. T. Giffin, and W. Lee. Automatic reverse engineering of malware emulators. In *IEEE Sec. and Privacy*, 2009.
33. A. Shaw. Livingsocial hacked: Cyber attack affects more than 50 million customers. `http://abcn.ws/15ipKsw`, April 2013.
34. V. Silveira. An update on linkedin member passwords compromised. `http://linkd.in/Ni5aTg`, July 2012.
35. W. T. Strayer, D. E. Lapsley, R. Walsh, and C. Livadas. Botnet detection based on network behavior. In *Botnet Detection*, 2008.
36. Symantec. Advanced persistent threats. `http://bit.ly/1bXXdj9`, December 2013.
37. R. Tian, L. Batten, and S. Versteeg. Function length as a tool for malware classification. In *IEEE MALWARE*, 2008.
38. Trend Micro. Trend Micro Exposes LURID APT. `http://bit.ly/18mX82e`, September 2011.
39. V. V. Vazirani. *Approximation algorithms*. Springer, 2004.
40. G. Yan, N. Brown, and D. Kong. Exploring discriminatory features for automated malware classification. In *DIMVA*, 2013.
41. H. Zhao, M. Xu, N. Zheng, J. Yao, and Q. Ho. Malicious executables classification based on behavioral factor analysis. In *IC4E*, 2010.
42. Y. Zhou and X. Jiang. Dissecting android malware: Characterization and evolution. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 95–109. IEEE, 2012.