Towards A Methodical Evaluation of Antivirus Scans and Labels

"If You're not Confused, You're not Paying Attention"

Aziz Mohaisen[1], Omar Alrawi[2], Matt Larson[3], and Danny McPherson[1]

[1] Verisign Labs
[2] Qatar Foundation
[3] Dyn

Abstract. In recent years, researchers have relied heavily on labels provided by antivirus companies in establishing ground truth for applications and algorithms of malware detection, classification, and clustering. Furthermore, companies use those labels for guiding their mitigation and disinfection efforts. However, ironically, there is no prior systematic work that validates the performance of antivirus vendors, the reliability of those labels (or even detections), or how they affect the said applications. Equipped with malware samples of several malware families that are manually inspected and labeled, we pose the following questions: How do different antivirus scans perform relatively? How correct are the labels given by those scans? How consistent are AV scans among each other? Our answers to these questions reveal alarming results about the correctness, completeness, coverage, and consistency of the labels utilized by much existing research. We invite the research community to challenge the assumption of relying on antivirus scans and labels as a ground truth for evaluating malware analysis and classification techniques.

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

1 Introduction

Antivirus (AV) companies continuously evolve to improve their products. AV products provide users with an added protection from malware threats, but they are not a complete solution. Malware evolves at a much faster rate than AV products, which then forces AV companies to innovate and improve approachs for malware detection. AV products provide two major functionalities: first and more importantly, detecting malicious software and secondly, labeling malware based on a family association. Labeling is an important problem to AV vendors because it allows them to filter known malware and focus on new malware families or variants of similar malware families. Labeling also allows the AV vendors to track a malware family and its evolution.

The AV market is very diverse and provides much room for competition, allowing new companies to build AV products and compete for a share of the market. This diversity of AV software vendors creates disorganization in the AV market characterized by a lack of standards for information sharing, malware family naming, and transparency. For example, each AV company has its own way of naming malware families as they are discovered. Malware names are usually created by analysts who study new malware samples, by utilizing artifacts within the malware to derive the given names. Some malware is so popular in underground forums, like SpyEye, Zeus, ZeroAccess, DirtJumper, etc., that AV vendors use those names given to the malware by the authors or the malware market. Other smaller and less prominent malware is usually named independently by each AV company. For example, targeted malware, which is known as advance persistent threat (APT), is low key malware that AV vendors name and track independently.

Malware naming and labeling has many useful applications in the security field. Security practitioners have an interest in identifying a malware by a family name so that they can mitigate the threat for their organization. AV vendors can use labels to filter out insignificant malware and focus only on high priority malware families. Researchers in academia have benefited from detections and labeling of malware provided by AV vendors in many ways. For example, 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. In fact, there exists a large body of academic literature that relies on AV labels to verify methods and techniques, including [2,3,6,12,13,16,20,22,23] (a survey of those works is in [15]).

However, the use of AV labels for validating classification research—while creating a ground truth for comparing different works to each other— has many shortcomings and pitfalls. First, oftentimes malware samples collected by researchers are not necessarily represented in their entirety within a single malware scanning engine. Accordingly, researchers tend to use multiple engines to cover their datasets, despite inconsistencies in labeling and naming conventions used by those engines. Those inconsistencies are resolved by translating names from one AV vendor to another, although—as we mentioned earlier—different AV vendors may use different names to refer to the same family. 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 label [9].

1.1 The Origins of Inconsistency and Implications

The AV inconsistencies arise because of the methods used by the AV vendors for detection. The primary goal for an AV vendor is detect malicious code, hence making labeling a secondary priority. Most AV vendors use a combination of static and heuristic-based methods to identify and label malware. Static signatures are insufficient because malware authors use obfuscation techniques, which quickly outdates a static signature. A heuristic signature, on the other hand, allows detection based on the behavior of the unknown software. A heuristic signature can consist of several behavior conditions that will trigger the signature. For example, if a piece of unknown software listens on a port for incoming connections, injects code into privileged processes, and hooks specific Windows

APIs, then a heuristic signature is triggered and an alert is generated. These heuristic rules are loosely designed to detect malicious code, hence they are not suitable for labeling due to their generic labeling schemes.

Those inconsistencies create inefficiencies in the industry and 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 $\mathcal{A}_i$ detects a malware with label ℓ_1, the user might have a mitigation plan for that malware family. On the other hand another AV vendor, $\mathcal{A}_j$, detects the same malware as ℓ_2, then the user of $\mathcal{A}_j$ 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 or reputation damage. Understanding the size of the damage caused by this issue is nontrivial, since companies are conservative in revealing 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. However, there has been recent public information that support, highlight the trend, and put good figures on that cost and pervasiveness of such phenomena; examples include the hacking of LinkedIn [18], Ubisoft [4], LivingSocial [17], and most famously Nissan [10].

An even worse problem related to those inconsistencies is when the same AV engine detects the same malware family with different labels due to an evasion technique used by the malware. For example, if a malware is initially detected using a static signature, then later—due to its polymorphism technique—heuristically using a generic malicious behavior, the AV vendor will give it another label. This will create 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. For example, mislabels may propagate error throughout an experiment that relies on those labels as a ground truth. In the literature, researchers widely accepted AV labels as a ground truth of malware family membership, including applications of classification, clustering, and alternative detection techniques.

Motivated by our recent work on cross-validating AV labels against highly-accurate expert-vetted and labeled malware samples, we pursue the study of systematically understanding those inconsistencies and reveal several interesting insightful remarks that greatly affect the way applications based on AV provided labels (or even detections) work. Our study to address the problem, while inspired by the work in [2], is the first of its type to go at length to systematically evaluate those labels and detections. We do this by considering one popular malware family, the Zeus banking Trojan, thus giving AV scanners many benefits of the doubt. The Zeus banking Trojan is a famous banking Trojan that is used by cyber criminals to run a botnet to steal money, credentials, and system resources from the infected victims. In particular, the Zeus source code was leaked in 2011 and since then there has been numerous variants that have surfaced [7]. Our samples are used so that their detection in our system is old enough to make sure that they are populated in those AV scanners. The popularity of the studied malware family makes it a good candidate to evaluate detections and labels, because one would expect this family to be well researched in the AV community, thus reducing inconsistencies in labels and detections across different AV vendors.

1.2 Contribution

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 a 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 out 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 section 4). 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 [2,3], 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[1].

1.3 Organization

The organization of the rest of this paper is as follows. In section 2 we provide an overview of the dataset we used in this study and the method we use for obtaining it. In section 3 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 4 we review the related work, followed by concluding remarks, open directions, and the future work in section 5.

2 Datasets

To evaluate the different AV vendors based on a common ground of comparison, we use the Zeus banking trojan, which has been identified manually by analysts. Our dataset consists of 1,000 samples, which is large enough to derive meaningful insights into the problem at hand and small enough to be manually vetted for correct results[2]. To identify the label of this family, we used forensic memory signatures to identify a set of possible Zeus samples from our malware repositories,

[1] Ironically, some of those works pointed out the problem and yet used AV-provided labels for validating their malware clustering algorithms [3,15].

[2] We use malware samples accumulated over a period of a year (mid 2011 to 2012). As we will see later, this would give the AV vendors an advantage and might overestimate their performance compared to more emerging threats (APT).

then we manually vetted the set to ensure our final data set is clean of malware families that might falsely trigger our memory signatures. More details on this method for vetting malware samples is described in [9]. For the evaluation of our data set we used VirusTotal [1] signatures for 48 AV engines to test several evaluation measures. We discarded all engines that provided detections for only less than 10% of our dataset.

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 we only use the AV vendor name and their detection label. VirusTotal will provide more AV results 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. However, because no researchers have had an alternative to the labels the AV scanners provide, so far the completeness, consistency, and correctness—the three comparison and evaluation measures we study in §3—of AV labels and scans were not challenged. Implications of those metrics of an AV scan were overlooked in the literature.

3 Measurements

3.1 Evaluation Metrics

In this work we use several evaluation metrics, namely completeness, consistency, and correctness, and coverage. The metrics are defined as follows:

– Completeness: For our references dataset $\mathcal{D}$, we compute the *completeness* of the scans of an AV vendor $\mathcal{A}_i$ as the number of detections returned by $\mathcal{A}_i$ normalized by the size of the dataset, $|\mathcal{D}|$.
– Consistency: The *consistency* is computed as the agreement (or disagreement) in detections between $\mathcal{A}_i$ and the rest of AV vendors we tested. For that, we use the Jaccard distance which captures the number of samples consistently detected or undetected by two vendors $\mathcal{A}_i$ and $\mathcal{A}_j$ and results in $n-1$ consistency values for $\mathcal{A}_i$.
– Correctness: Given our reference dataset $\mathcal{D}$, we compute the *correctness* of an AV vendor $\mathcal{A}_i$ as the number of detections matching our reference detection normalized by the size of the reference dataset, $|\mathcal{D}|$.
– Coverage: We define the *coverage* as the number of the AV scans (list of detections and labels, respectively) required for achieving a correct and complete scan. This is, as various AV scans will provide partial correctness and completeness scores for the reference dataset, this measure would determine how many AV vendors one needs to utilize to get a perfect score for both measures.

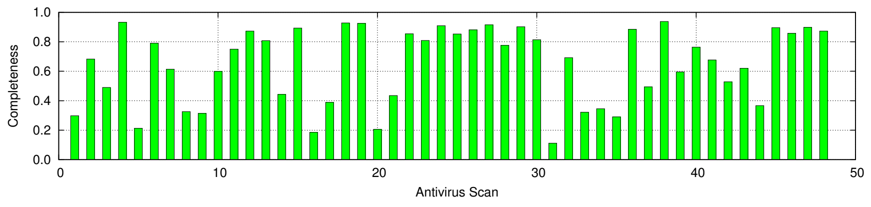

Fig. 1. Completeness of detections by 48 AV vendors. Popular AV scans used in the literature include Avira (Antivir), Kaspersky, Symantec, McAfee, Microsoft, AhnLab, AVG, and ClamAV, which are bars numbers: 27, 18, 11, 4, 36, 40, 47, and 17, respectively. (in all figures)

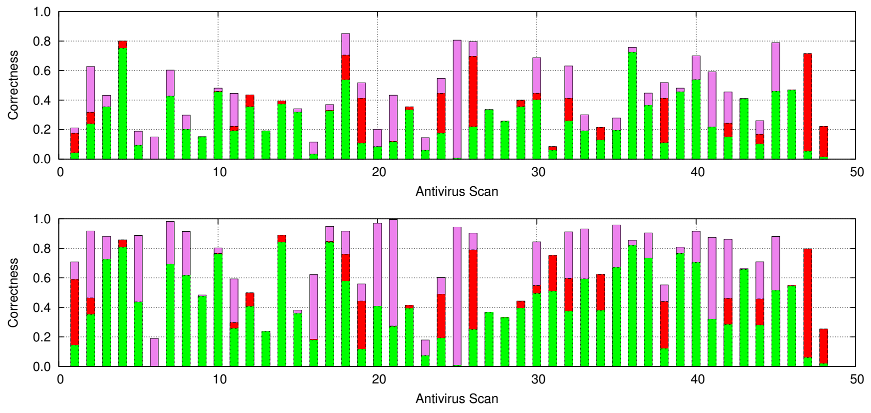

Fig. 2. Correctness of detections by 48 vendors. Color code: green is precise detection (Zbot), red and violet are translated detections (generic and trojan). Upper figure is normalized by 1000 (incorporates the completeness score), while the lower figure is normalized by number of detections of the given AV scan.

3.2 Results and Analysis

Completeness. For completeness, and as explained above, we use the ratio of detections out of our 1,000 sample set per an AV engine. For example, if an AV engine $\mathcal{A}_i$ has 950 detections for the 1,000 sample dataset then AV engine $\mathcal{A}_i$ has a 0.95 completeness regardless to what labels that are returned by the named AV. Figure 1 plots the completeness of all AV engines that were used in the evaluation. The results show a detection rate as low as 0.15 for our dataset and as high as 0.94 for the given AV engines. Roughly 45% of the AV engines in our study have a less than 70% detection rate (0.7 completeness score).

While the completeness scores given to AV scanners are not surprising, the high diversity in the score and large range are. Thus, the result has many interesting implications. Most importantly, given that no AV scanner provided a completeness of one, we cannot establish consistency over a labeling based on a single reference AV vendor. Furthermore, we cannot establish a consensus on what different AV providers mean by their labels for the entire dataset. Second, even the most widely used AV scanners are do not provide sufficient results: among the most widely used AV vendors in the academic community for validation of studies are Avira (Antivir), Kaspersky, Symantec, McAfee, Microsoft, AhnLab, AVG, and ClamAV. While they are among the most complete AV scans for the reference dataset (with completeness scores ranging from 0.76 to 0.94), we notice that the small "incompleteness" of scans is problematic to the application for which the scans are used. While researchers strive to achieve 99% of accuracy in their classification of malware samples [3], a 6% 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.

Correctness. We define the correctness as the ratio of detections with a correct label per AV engine for all of our dataset. In our study we considered three labels to be acceptable for correctness, "Zbot," "Trojan," and "Generic." Although those labels might be anticipated to yield high results, especially for generic labeling (Generic) the results show otherwise. Figure 2 illustrates two plots of the correctness evaluation. The bar chart is color coded differently for each label mentioned above, green is "Zbot," red is "Generic," and violet is "Trojan" (the height of the bar is inclusive of all detections and variants). The difference between the two bar charts is the normalization value used for both. While the first bar chart is normalized based on the size of our dataset of 1,000 samples to incorporate the completeness score, the second bar chart is normalized by the number of detections for the given AV engine. We observe that the majority of AV engines label the malware "Zbot" less than 50% (0.5 of correctness) of the time per detection. We also observe that the "Zbot" label is used by almost all of the AV engines. Note that the correctness score for each AV engine is lower than the completeness score because other labels are used by the AV engine to label our dataset, which vary outside the three major labels we chose.

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 label, and thus a labeling provided by an AV vendor cannot be used as a certain ground truth for labeling.

Consistency. We define the consistency as an AV engine detecting a malware sample alongside other AV engines. The consistency is determined per sample and is compared across all AV engines. We observed on average an AV engine is about 50% 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 3 illustrates the consistency of each AV engine across all other engines using box plots (min, first quartile, median, third quartile, and max). 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.

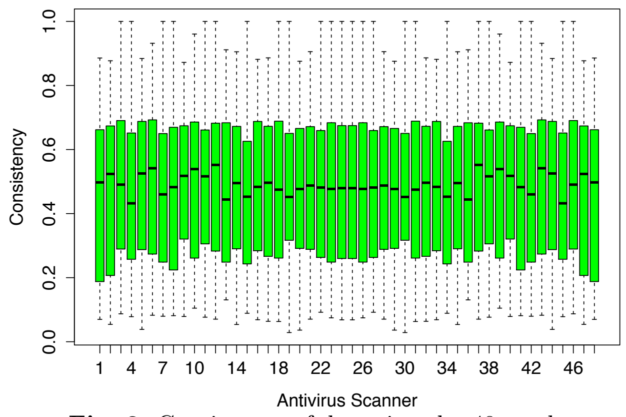

Fig. 3. Consistency of detections by 48 vendors.

Coverage: How many AV scanners? For completeness and correctness we pose the following question: What is the least number of scanners it takes to get a complete (or correct) scan of our reference dataset? This question stems from the fact that many researchers in the literature combined more than one AV vendor's results to get a better coverage of labels for their experiments. Answering this question is not as easy as it sounds: the problem is the optimization version of the set-cover problem, which is known to be NP-hard. Thus, we consider heuristics to answer the question.

Again, we use the same scans we used for plotting the previous figures of the completeness and correctness scores. We use two strategies for each score, namely l/s and s/l, as shown in Figure 4. For l/s, we start by considering potential scans to obtain the required completeness from large to small, add them to the list of final scans, increase the number of AV scanners by one for each, and recompute the completeness (correctness) score respectively. The s/l strategy does the opposite. We use the strategies as extremes, and don't consider the best strategy [3]. As expected, we notice that it takes fewer scanners to achieve a completeness of 1 than it takes to achieve correctness of 1, even with the better-performing strategy. Numerically, we observe that while 5 scans are required to achieve completeness of 1, 22 AV scans give only 0.97 of correctness. Indeed, even 48 AV scanners (the total) were able to achieve only 0.99 of correctness.

4 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 clas-

[3] The greedy strategy, by adding the AV scan with least overlap to the current set, is the best known approximation [21].

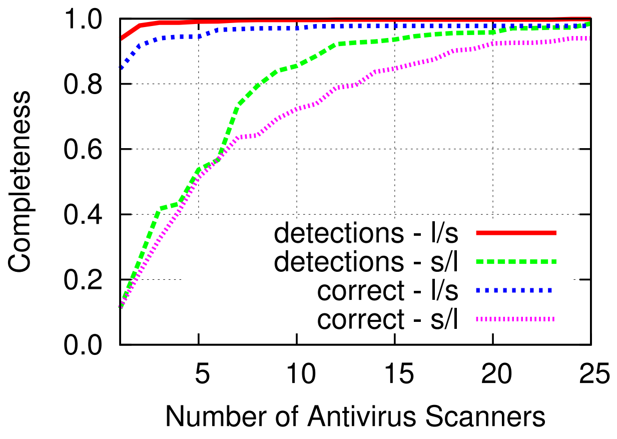

Fig. 4. Completeness (per detections and correct detections) when using largest to smallest and smallest to largest scan to accumulate the entire dataset.

sification and analysis [2, 3, 5, 8, 9, 11–14, 19, 20, 23], there is less work done on understanding the nature of those labels. Recent works, like [15] and [13] only pointed out the problem of AV-provided labels without any measurements or evaluation.

To the best of our knowledge, the only prior work dedicated to systematically understanding AV-provided labels is due to Bailey et al. [2]. 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 [2] relies on an AV vendor as a reference and compares other vendors to it. In particular, the authors use McAfee as the presumed complete and accurate reference of detection and labeling and compare a handful other vendors to it. Our technique avoids this issue by relying on a manually inspected reference set against which the performance of many AV vendors is tested.
– 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: the correctness, completeness, and coverage. On the other hand, the prior work considers all results provided by the reference AV scan to be correct and compares other AV scans to them.

5 Conclusion and Future Work

In this work, we unveil the danger of relying on incomplete, inconsistent, and incorrect malware labeling systems provided by AV scanners and using them in the research community for validating malware analysis and classification techniques. Our study shows that one needs many independent AV scanners to obtain complete and correct labels. 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 Zeus malware family, since we didn't use all samples known to be Zeus against the AV scanners. Our study is, however, meaningful in answering the limited context's questions it poses for 1000 malware samples. 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.

To this end, in the future we will try to answer those questions with more manually-vetted malware samples belonging to different families, and by studying better ways of obtaining consensus over AV-provided labels, ways that can tolerate many inconsistencies among vendors. We see a solution to the problem by enabling information sharing, so one of our future works is to explore how this sharing would enable better use of indicators for better malware labeling. We will release all datasets used in this study (AV scans, hashes, and scripts used for comparison), to help pursue alternatives. We hope this note will trigger further investigation and attention in the community to this crucial issue.

Acknowledgement. We would like to thank Andrew West for proofreading this work, and Allison Mankin and Burt Kaliski for their feedback on an earlier version of this work.

References

1. —. VirusTotal - Free Online Virus, Malware and URL Scanner. https://www.virustotal.com/en/, August 2013.
2. M. Bailey, J. Oberheide, J. Andersen, Z. Mao, F. Jahanian, and J. Nazario. Automated classification and analysis of internet malware. In *RAID*, 2007.
3. U. Bayer, P. M. Comparetti, C. Hlauschek, C. Krügel, and E. Kirda. Scalable, behavior-based malware clustering. In *NDSS*, 2009.
4. D. Kerr. Ubisoft hacked; users' e-mails and passwords exposed. http://cnet.co/14ONGDi, July 2013.
5. J. Kinable and O. Kostakis. Malware classification based on call graph clustering. *Journal in computer virology*, 7(4):233–245, 2011.
6. 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.

7. P. Kruss. Complete zeus source code has been leaked to the masses. http://www.csis.dk/en/csis/blog/3229, March 2011.
8. A. Lanzi, M. I. Sharif, and W. Lee. K-tracer: A system for extracting kernel malware behavior. In *NDSS*, 2009.
9. A. Mohaisen and O. Alrawi. Unveiling zeus: automated classification of malware samples. In *WWW (Companion Volume)*, pages 829–832, 2013.
10. New York Times. Nissan is latest company to get hacked. http://nyti.ms/Jm52zb, April 2013.
11. Y. Park, D. Reeves, V. Mulukutla, and B. Sundaravel. Fast malware classification by automated behavioral graph matching. In *CSIIR Workshop*. ACM, 2010.
12. R. Perdisci, W. Lee, and N. Feamster. Behavioral clustering of http-based malware and signature generation using malicious network traces. In *USENIX NSDI*, 2010.
13. K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov. Learning and classification of malware behavior. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 108–125, 2008.
14. 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.
15. 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.
16. M. I. Sharif, A. Lanzi, J. T. Giffin, and W. Lee. Automatic reverse engineering of malware emulators. In *IEEE Sec. and Privacy*, 2009.
17. A. Shaw. Livingsocial hacked: Cyber attack affects more than 50 million customers. http://abcn.ws/15ipKsw, April 2013.
18. V. Silveira. An update on linkedin member passwords compromised. http://linkd.in/Ni5aTg, July 2012.
19. W. T. Strayer, D. E. Lapsley, R. Walsh, and C. Livadas. Botnet detection based on network behavior. In *Botnet Detection*, 2008.
20. R. Tian, L. Batten, and S. Versteeg. Function length as a tool for malware classification. In *IEEE MALWARE*, 2008.
21. V. V. Vazirani. *Approximation algorithms*. Springer, 2004.
22. G. Yan, N. Brown, and D. Kong. Exploring discriminatory features for automated malware classification. In *Proceedings of the 10th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, 2013.
23. H. Zhao, M. Xu, N. Zheng, J. Yao, and Q. Ho. Malicious executables classification based on behavioral factor analysis. In *IC4E*, 2010.