RC25288 (WAT1205-073) May 23, 2012
Physics

IBM Research Report

Towards Consensus Labeling of Malware Threats

Ting Wang, Xin Hu
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598
USA

Towards Consensus Labeling of Malware Threats

Ting Wang Xin Hu
IBM T.J. Watson Research Center

ABSTRACT

The unprecedented immensity and variety of malware threats (e.g., virus, Trojan horses, worms) have spurred intensive research on large-scale malware analysis in both academia and industrial communities; yet, the knowledge bases built by such effort have not been collectively leveraged to a large extent. One fundamental barrier facing the integration of threat intelligence is the lack of malware labeling standards. We show the severity of this problem by an in-depth empirical study of the labeling systems of five popular anti-virus engines using a large collection of malware instances. Instead of attempting to unify the malware naming conventions, we propose a pragmatic alternative: leveraging correspondence evidences from multiple anti-virus sources to create a virtual, consensus malware categorization, such that different anti-virus vendors can communicate through this consensus scheme without changing their local naming conventions. We present a prototype malware label matching system LATIN that makes it possible to tell whether two malware samples under different naming conventions refer to the same malware category simply by their names.

1. INTRODUCTION

The defense against malware (such as virus, worms, Trojan horses, spyware, rootkits, and backdoors) has been a prominent topic for computer security research for decades. Nevertheless the challenge has never been immenser due to the unprecedented scale and variety of threats: new malware samples are now created at a rate of millions per day and infiltrate to every new platform [17], which goes way beyond the defense capacity of any single organization.

1.1 Malware Threat Intelligence

Over the past decades, the research efforts on large-scale malware analysis from both academia and industrial communities have built up a large number of malware threat knowledge bases, with abundant information such as threat labels, behavior descriptions (e.g. Kaspersky's SecureList), and binary signatures. The techniques of collecting such intelligence have evolved from manually creating threat description libraries by human analysts to developing malware classification systems that automatically extract distinguishing features and categorize new samples. Today a modest AntiVirus (AV) engine is typically shored up by a knowledge base about millions of threats. Intuitively these knowledge bases feature different coverage and expertise for parts of the malware universe (details in Section 2); it is thus beneficial to collectively exploit such intelligence. In practice, however, the integration of malware threat intelligence is still limited to fairly small scales (e.g., [7]) due to a number of great challenges (e.g., the lack of standards and the conflict of interests).

In this work we focus on one of such fundamental barriers: the lack of malware labeling standards. The labeling system of a malware knowledge base essentially specifies the categorization of its malware collection, i.e., if two malware should be considered as similar. The difference of two labeling systems reflects the discrepancy of the underlying categorization. As a concrete example, consider the labels given to a set of malware samples by three popular AV engines, Kaspersky, NOD32 (ESET), and BitDefender, as shown in Table 1.

One can observe many types of discrepancies in the three labeling systems:

- *Coverage* (Section 2.1): Both Kaspersky and BitDefender consider S_1 as malicious, while NOD32 fails to detect it.
- *Granularity* (Section 2.2): Both Kaspersky and NOD32 give detailed categorization for S_2, while BitDefender only reports it as generic malware.
- *Categorization* (Section 2.3): NOD32 considers S_3 and S_4 as belong to *SdBot*, while Kaspersky categorizes them as two different categories, *Trojan-Proxy* and *Backdoor*. Meanwhile, S_3 and S_4 are also identified as *Bot* by BitDefender; however, BitDefender assigns a more generic label to S_3.

Clearly addressing the discrepancy in different malware labeling systems (or in other words, achieving their "consensus") would be a crucial step towards the integration of malware threat intelligence. Moreover, it would benefit a variety of applications in its own right. Consider dataset comparison as a concrete example: due to the lack of malware classification benchmarks (the luxury enjoyed by some other research communities, e.g., machine learning [18]), new malware classification systems are often evaluated against self-composed threat datasets using certain knowledge bases, resulting in hard-to-compare results.

	Kaspersky	NOD32	BitDefender
S_1	Trojan-Downloader.Win32.WinShow	N/A	Trojan.Downloader.WinShow
S_2	Trojan-Spy.Win32.Agent	Win32/Spy.Agent	Trojan.Generic
S_3	Trojan-Proxy.Win32.Ranky	Win32/IRC.SdBot	Backdoor.Bot
S_4	Backdoor.Win32.SdBot	Win32/IRC.SdBot	Generic.SdBot

Table 1: Malware labels assigned by three AV engines.

	Precision	Recall	# of Samples in reference dataset	# of Classes in reference dataset
System 1 [11]	0.932	0.928	3,935	200
System 2 [10]	0.80	0.35	4,821	20

Table 2: Accuracy of malware classification and characteristics of test datasets (Precision measures how well the individual clusters agree with the original classes and Recall R measures how much the malware classes are scattered across the clusters. When evaluated against the labeling scheme that breaks down input data into a large number of classes, the clustering/classfication system will achieve better performance, especially Recall scores.

Table 2 shows the clustering accuracy and the test malware datasets as reported by two malware classification systems [11, 10]. To interpret such results requires to compare the test malware samples (dataset alignment), while such executable instances are often unpublishable for their sensitivity and many privacy concerns. However, with the help of the consensus labeling system, it is possible to contrast the datasets provided only the labels of the samples under any specific labeling systems (which is much less sensitive comparing with the binary executables) are available.

1.2 Our Contributions

To our best knowledge, this work represents the first attempt of systematically addressing the discrepancies in different malware labeling systems in a large scale.

We start with investigating the severity of this problem by conducting an empirical study of the current state of malware labeling systems (Section 2). Specifically, we apply five popular AV engines in the market (Kaspersky, Avira, Avast, NOD32, BitDefender) [6] over a large collection of malware executable instances (over 250K) and present in-depth analysis of the diagnosis results to understand the discrepancies in different threat knowledge bases.

More importantly, we develop a novel threat label matching system, Label Alignment and MaTchINg System (LATIN) (Section 3). Figure 1 sketches its architecture, which comprises two main components, *label aligner* and *label matcher*. Label aligner takes a collection of malware instances (and/or the alias information if available in the threat knowledge bases) as input and automatically extracts the "consensus" categorization of the labeling systems underlying these knowledge bases. Equipped with the consensus categorization, label matcher is able to tell whether two malware labels (with optional input including specification, instances, etc.) in two different knowledge bases refer to the same category of malware.

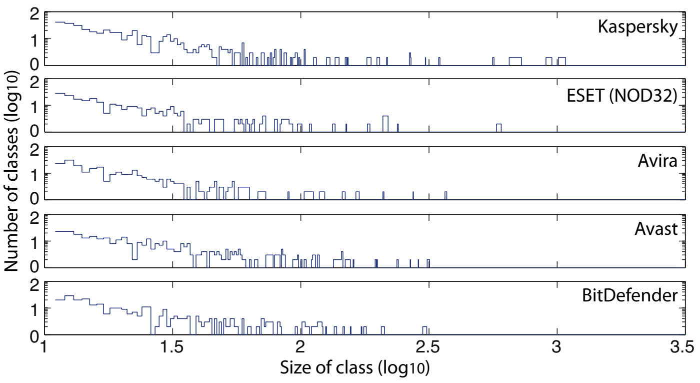

Figure 2: Distribution of class sizes (Logarithm scale) under different labeling systems.

We empirically evaluate the efficacy of LATIN using the alias information available in the threat knowledge bases (Section 4). The result is encouraging: LATIN successfully matches over 74.5% of alias simply based on the label information.

2. STORY OF CHAOS

The first step of addressing the discrepancies of malware labeling is to understand how these discrepancies look like. We select five anti-virus engines (Kaspersky, Avira, Avast, NOD32, BitDefender) popular in market and feed them with a large collection of malware samples (235,947 distinct instances). We intend to understand the labeling discrepancy via analyzing the reported diagnosis results.

We notice that except for generic labels (e.g., Trojan-Generic) and heuristic diagnosis, most malware labels provided by these anti-virus engines consist of four or less fields: $\langle type \rangle$ (e.g., Backdoor), $\langle platform \rangle$ (e.g., Win32, which is missing in Avira), $\langle family \rangle$ (e.g., Ceckno), and $\langle vairant \rangle$ (e.g., crs, which is missing in BitDefender and NOD32). We thus use the combination of $\langle type \rangle$+$\langle family \rangle$ to specifie the *class* of malware.

Figure 2 illustrates the distributions of class sizes under the five labeling systems. The fairly similar patterns indicate that the concept of malware class prevails in different labeling systems, which can thus serve as the basis of comparing different labeling systems.

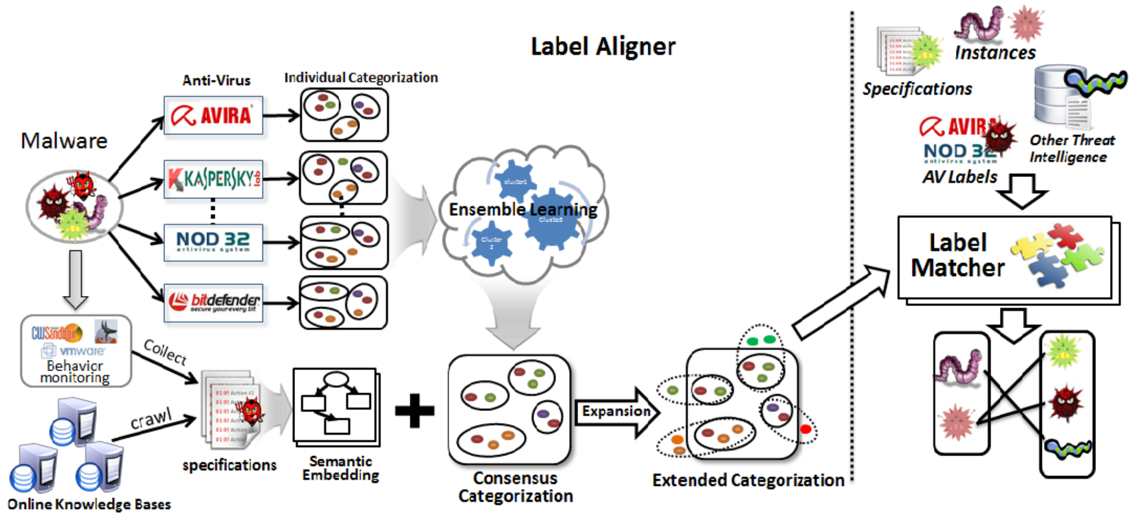

Figure 1: Overall architecture of LATIN.

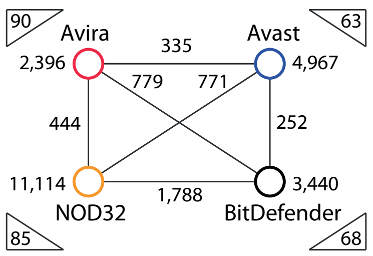

Figure 3: Number of samples undetected by AV engines.

AV engine	generic label	number of samples
BitDefender	Application.Generic	533
	Trojan.Generic	**46,791**
	IRC-Worm.Generic	268
	Adware.Generic	112
	Backdoor.Generic	6,059
	Dialer.Generic	350
Avast	Win32:Rootkit-gen	2,684
	Win32:Trojan-gen	**52,703**
	Win32:Spyware-gen	4,982
	Win32:Malware-gen	9,591

Table 3: Generic labels assigned by AV engines.

We are now ready to tell the story of chaos.

2.1 Coverage

The first phenomenon we notice is that different anti-virus engines (as supported by the underlying knowledge bases) demonstrate drastically different coverage in terms of malware detection.

Figure 3 illustrates the number of malware samples that are not detected by different AV engines, where the number by each node represents the total samples missed by that engine, the number on each edge represents the samples missed by both engines at the ends of the edge, while the number in each triangle is the samples undetected by the three engines at that corner.

It is noticed that while individual AV engines may miss a large number of malware population (e.g., NOD32 fails to detect 4.71% of the entire malware collection) due to their specific coverage and expertise, the coverage increase by combining different AV engines is drastic (e.g., the samples undetected by Avira, Avast, and BitDefender is below 0.03%). Particularly there are no samples missed by all four engines, which again highlights the necessity of threat intelligence integration.

2.2 Granularity

The second type of discrepancy we notice in the diagnosis results reflects in the granularity of malware classes assigned by different engines. For example, in Table 1 BitDefender gives S_2 a much more generic label than NOD32. Table 3 summaries the generic labels assigned by AV engines (only the generic labels which are assigned to more than 100 samples are listed). It is noted that 19.8% and 22.3% of samples are labeled as *Trojan.Generic* and *Trojan-gen* by BitDefender and Avast, respectively, and 4% of samples are labeled as *Malware-gen* by Avast. Clearly such generic labels are non-informative to differentiate two malware samples simply by their names.

2.3 Categorization

In the following we focus on non-generic labels assigned by AV engines, i.e., labels that can assign the malware samples to specific malware classes. Thus for a given malware collection, a labeling system essentially specifies one way of categorizing the samples in the collection; in other words, whether two samples are considered as belong to the same

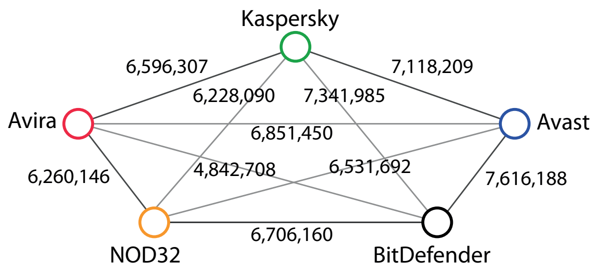

Figure 4: Number of inconsistent pairs in two labeling systems.

class. The third type of discrepancy reflects in the difference of such categorization. Figure 4 summarizes the number of pairs classified into one class by one AV engine (the node closest to the number) but regarded as of different classes by another (the node at the other end of the edge). It is noticed that any two AV engines disagree on 3∼7 million sample pairs.

3. SYSTEM LATIN

Figure 1 illustrates the system architecture of LATIN. The goal of LATIN is to leverage threat intelligence from multiple data sources to derive a consensus categorization which allows us to accurately match between diverse naming/labeling systems from different vendors or intelligence providers. LATIN is composed of two components: 1) *Label Aligner*, responsible for processing input threat intelligence and reconcile the sample labels into an extended categorization via a scalable ensemble learning algorithm; and 2) *Label Matcher*, capable of finding correspondence between different labels generated from different threat intelligence.

Given a large number of malware samples, Label Aligner works in following three phases, i.e, Probing, Categorization and Expansion, each of which we elaborate below.

- **Phase 1: Probing** LATIN first probes the various knowledge bases to find a set of instance-level correspondence of different labeling systems. A knowledge base can be any data source that provide categorization information for the malware instance. Examples include Anti-virus software (which provide family names), malware analysis frameworks (e.g. Anubis[2] which provide behavior profiles), online virus encyclopedia from many AV vendors (which provide malware specifications). This flexibility allows LATIN to be easily extended to accommodate additional intelligence sources.

- **Phase 2: Categorization** LATIN creates individual categorization based on information collected from each intelligence. For example, LATIN parses malware labels from a particular vendor identifying malware samples that share the same family name and grouping them into the same cluster. The ensemble learning component takes as input the "local" categorization of these malware instances and aggregates the instance-level correspondences between local categorizations to identify the consensus categorization. One challenge of extracting such consensus categorization is the scalability issue. For example, using our malware collection, we are facing over 235K malware samples and over 25K malware classes. Conventional ensemble learning algorithms that rely on techniques such as spectral graph clustering [14] is too slow for our task. We develop a novel consensus learning algorithm based on power iteration clustering [13], which reduces the running time of ensemble learning from days to hours in our case.

- **Phase 3: Expansion** LATIN integrate additional information available in the knowledge base to expand the instance-level or class-level correspondence beyond the instances available in the input dataset. For example, many vendors maintain a correlation between their name and used by other vendors in their malware description databases or virus encyclopedia (often in the 'Alias' section). This provides useful information absent of the particular instances. LATIN also performs text mining (i.e. semantic embedding via tf-idf weighting) on crawled malware specifications, attempting to match specifications of malware that are not in the input dataset with specifications of input samples. Via this step, LATIN is able to expand to malware samples that are otherwise impossible to include in the consensus categorization due to the lack of their binaries.

Through the second component *Label Matcher*, LATIN allows users to query the correspondence between information obtained from different threat intelligence (e.g. malware description, labels from different AV vendors and clustering/classification results) . Label Matcher uses the the *Extended Consensus Categorization* (ECC) created by the Label Aligner as the reference and maps the input threat intelligence to the closest category in the ECC, such that a correspondence relationship can be identified if the information of two input malware threats are mapped to the same consensus category. Notice that the mapping is often one-to-many as oppose to one-to-one mappings in the traditional labeling system (where a malware program only has one unique label and thus belongs to a single family). This is caused by the conflicting opinions from different vendors where some of them may divide the set of samples into multiple groups whereas others may prefer lumping them together. Hence, in such cases, *Label Matcher* will provide a probability matching score indicating the *likelihood* that the sample belongs to a particular category. Notice that the uncertainty can be reduced if additional intelligence about this sample is also available (e.g. names from more AV vendors, its behavior specification) because such additional information provide additional dimension in searching for the closest consensus category.

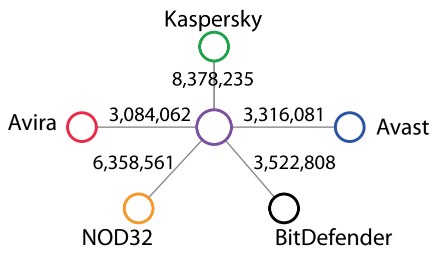

Figure 5: Number of inconsistent pairs between consensus and individual categorization.

4. PRELIMINARY EVALUATION

Next we present an empirical evaluation of Latin by using it to match labels under different labeling systems. While the design of Latin allows it to be augmented by additional information (e.g., threat description, alias, features) in threat knowledge bases (Section 3), to make the task challenging, here we only rely on the collection of malware samples (235,947 distinct instances) as input to train the label aligner and use the alias information as ground truth for evaluating label matcher.

We first validate the quality of the consensus categorization generated by label aligner. We consider good consensus categorization should have as few discrepancies with each individual categorization as possible. We therefore measure the number of inconsistent pairs between the consensus and each individual categorization, with results shown in Figure 5. Compared with the number of inconsistent pairs between two individual categorizations (shown in Figure 4), the consensus categorization reaches the maximum agreement among all the individual categorizations, thus representing a better way of categorizing the malware collection.

We then apply label matcher to match the labels under different labeling systems. We use the alias entries (each entry listing the "also-known-as" names under other AV engines) available in the threat knowledge bases as the test dataset. The dataset consists of total 12,411 entries and the number of intersection with our malware sample collection is 1,103.

For each pair of AV engines, we randomly select 1,000 pairs of aliases and test if Latin maps them to a same class in the consensus categorization. It is noted that there are cases that one class C in individual categorization is split and mapped separately to multiple classes $\{C'\}$ in the consensus categorization (note that one can set a mapping probability proportional to the number of samples in C mapped to each C'). In these cases, we consider the two aliases are "partially" matched if their corresponding classes in the consensus categorization overlap. Table 4 summarizes the results as categorized into complete match, partial match, and missed match. It is noticed that the overall accuracy (complete + partial match) is above 74.5% for all the cases. We expect that augmented with additional information (e.g., threat specification) and more training samples, the accuracy of Latin can be further improved.

pair of AV engines	complete	partial	miss
Kaspersky-Avira	439	320	241
Avira-Avast	534	211	255
Avast-BitDefender	617	204	179
BitDefender-Kaspersky	452	379	169

Table 4: Matching results: complete match, partial match, and missed match.

5. RELATED WORK

In addition to its unprecedented immensity and complexity, the current landscape of malware threats also features a huge discrepancies in the names used by different anti-virus product to ID a malware sample and an apparent lack of consensus among AV vendors in a common naming standard. Security communities have been debating this name inconsistency for decades long [5, 9, 3, 12] and several previous research also studied the severity for such problem as well as the potential root causes. Bailey et al [3] measured the consistency in terms of the ability of AV vendors to identify similar/identical malware in the same way and found that consistency is never a design goal of many AV systems – they assign the same label to identically behaved malware samples only the 31% to 61% of the time. Bureau and Harley analyzed the number of detection strings generated when scanning the same set of related samples and found a large variation in the number of labels used by different AV vendors from 100 to nearly 700. More recently, Maggi et. al [15] quantitatively showed that high degreeof inconsistency in terms of naming distance and scatter score exist across different vendors. Many reasons contribute to the malware naming mess. The first and foremost is the exponentially escalated number of new samples created nowadays [12, 4, 9]. Numbers published by different AV vendors constantly reports that hundreds of thousands to millions of new samples are received on a daily basis, making it practically impossible to examine every new malware program and agree on a reasonable name. In addition, AV vendors are under considerable pressure to push out the response/detection signatures promptly and wasting cycles on naming is always at the bottom of their priority list. Second, antivirus systems increasingly rely on generic signatures and other heuristics (behaviors, rare packers,etc), which are often unique to vendors, to identify malware. Methods for naming the detected malware samples thus vary across AV companies and usually do not follow any convention. For instance, sometimes when the malware does not have a name before, something catchy from the code or simply a random name might be picked to get it into the definition as fast as possible [12] and the cost is often too expensive to correct/change the names afterwards [4].

Several efforts have been made trying to remedy the naming confusion. The first attempt is the 1991 New Virus Naming Convention from CARO (Computer Antivirus Researchers

Organization) [19], followed by a series of proposals for revision [16, 8, 4]. These work attempt to standardize the naming convention in a well defined format e.g. Platform/FamilyName. GroupName.MajorVariat.MinorVariant and prohibit the use of specific information such as names of companies, brands and living people. Unfortunately, most of the major AV vendors decided to ignore these proposal. Even if they did, the name inconsistencies would still exist, as different vendors may assign different names to the same malware family/subfamily/variants. VGrep [20] and virustotal [1] offer pragmatic ways for users to search names from different vendors for a given malware instance, yet providing no systematic approaches to reconcile the inconsistent or even conflicting names. To our best knowledge, this work is the first attempt to systematically resolve the malware naming discrepancies by leveraging evidences of correspondences from multiple sources. This work aims to serve as first step to derive a common-ground benchmark that can be leveraged by the security research community to evaluate and compare results from different malware analysis systems.

6. CONCLUSION

Despite a series of effort to standardize the malware naming convention, the inconsistency in malware labels remains one fundamental barrier facing the integration of threat intelligence. Instead of attempting to unify the naming conventions by different AV vendors, we consider a pragmatic alternative: leveraging evidences from multiple AV sources to find a virtual, consensus naming scheme such that different sources can now communicate through this consensus scheme without changing their local naming conventions. We present a prototype system LATIN which demonstrates that this approach is promising for addressing the issue of label discrepancy to certain extent.

7. REFERENCES

[1] VirusTotal. https://www.virustotal.com/.

[2] Anubis. Analyzing unknown binaries. http://anubis.iseclab.org/, 2010.

[3] M. Bailey, J. Andersen, Z. M. mao, and F. Jahanian. Automated classification and analysis of internet malware. Technical report, In Proceedings of Recent Advances in Intrusion Detection, 2007.

[4] V. Bontchev. Current status of the caro malware naming scheme. http://www.people.frisk-software.com/ bontchev/papers/naming.html.

[5] P.-M. Bureau and D. Harley. A dose by any other name. In *Virus Bulletin conference Proceedings*, 2008.

[6] CNET. Most popular security software: http://www.cnet.com.au /software/security/most-popular.htm, 2012.

[7] Damballa. Integration partners: http://www.damballa.com/solutions/integration_partners.php.

[8] N. FitzGerald. A virus by any other name: Towards the revised caro naming convention. In *Proc. AVAR'2002 Conf., Seoul*, pages pp. 141–166., 2002.

[9] D. Harley. The game of the name malware naming, shape shifters and sympathetic magic. In *3rd International Conference on Cybercrime Forensics Education & Training*, 2009.

[10] X. Hu and K. G. Shin. Mutantx-s: Scalable malware clustering based on static features. In *CSE-TechReport University of Michigan*, 2011.

[11] J. Jang, D. Brumley, and S. Venkataraman. Bitshred: feature hashing malware for scalable triage and semantic analysis. In *Proceedings of CCS'11*, 2011.

[12] T. Kelchner. The (in)consistent naming of malcode. *Computer Fraud & Security*, 2010(2):5–7, Februrary 2010.

[13] F. Lin and W. W. Cohen. Power iteration clustering, 2010.

[14] B. Luo, R. C. Wilson, and E. R. Hancock. Spectral clustering of graphs. In *Proceedings of the 4th IAPR international conference on Graph based representations in pattern recognition*, GbRPR'03, pages 190–201. Springer-Verlag, 2003.

[15] F. Maggi, A. Bellini, G. Salvaneschi, and S. Zanero. Finding non-trivial malware naming inconsistencies. In *Proceedings of the 7th International Conference on Information Systems Security (ICISS)*, volume 7093 of *Lecture Notes in Computer Science*, pages 144–159. Springer-Verlag, 2011.

[16] G. Scheidl. Virus naming convention 1999 (vnc99). http://members.chello.at/erikajo/vnc99b2.txt, 1999.

[17] Symantec. Internet security threat report: http://www.symantec.com/threatreport, 2011.

[18] UCI. Machine learning repository: http://archive.ics.uci.edu/ml/.

[19] D. A. S. Vesselin Bontchev, Frierik Sklason. Virus naming scheme. ftp://ftp.informatik.uni-hamburg.de/pub/virus/texts/tests/naming.zip, 1991.

[20] I. Whalley. VGrep. http://www.virusbtn.com/resources/vgrep/index.