An Architecture Utilizing the Crowd for Building an Anti-virus Knowledge Base

Nguyen Hoang Thuan[1, 2], Pedro Antunes[1], David Johnstone[1], Minh Nhat Quang Truong[2]

[1] School of Information Management, Victoria University of Wellington, PO Box 600, Wellington, New Zealand
{Thuan.Nguyen, Pedro.Antunes, David.Johnstone}@vuw.ac.nz

[2]Can Tho University of Technology, 256 Nguyen Van Cu Street, Can Tho city, Vietnam
{nhthuan, tmnquang}@ctuet.edu.vn

Abstract. Recently, the behaviour-based technique was received attentions for its ability to detect unknown viruses. However, the literature suggests that this technique still needs to be improved due to high false-positive rates. Addressing the issue, the current work-in-progress proposed an architecture utilizing the crowd for building an anti-virus knowledge base, which considers not only virus behaviour but also behaviour from the new applications. This architecture also utilized anti-virus experts in the crowd for classified objects that are unclassified by machines. Using the classified objects, it used a machine learning algorithm to analyse application behaviour from the crowd for updating the knowledge base, and thus the corresponding anti-virus system can correctly diagnose and classify objects, reducing the false-positive rates.

Keywords: Anti-virus, Behaviour-based detection technique, Business process crowdsourcing, Crowdsourcing, Knowledge Base, Machine learning.

1 Introduction

Most computer users rely on anti-virus software to protect their computers from computer viruses. Although anti-virus software is offered by different vendors, it seems that the existing anti-virus software does not effectively protect the users. A recent study on six anti-virus products shows that "the anti-virus software doesn't always block malware from performing code injection" (p. 69) [1]. This requires an improvement of anti-virus strategies, especially an improvement on detection techniques. The call for improving existing detection techniques was recommended by several researchers [1, 2], who believe that the current detection methods are not enough powerful against the evolutionary of viruses.

By and large, two detection techniques have been deployed in anti-virus systems: signature-based detection and behaviour-based detection [1, 3]. The signature-based techniques, which are mainly based on the known threats' signature, can only recognize viruses from their known datasets and may have problems when viruses use obfuscation or polymorphism techniques [4]. Because of the fundamental constraints, the behaviour-based techniques have received the focus of recent studies [3-5]. Some advantages of the behaviour-based approach have been highlighted in the literature. For instance, it can detect unknown computer viruses and new variants of existing viruses [5-7]. It can also avoid processing a huge number of virus samples [4], and adapt over time by using machine learning algorithms [8, 9].

Although the benefits of behaviour-based techniques are well recognized, this approach still has problems for distinguishing malicious from regular behaviours [8]. In particular, the behaviour-based approach may lead to two types of errors. The first type currently considered too high [1] is false-positive detection, where benign applications are classified as malicious. For instance, Unikey, a Vietnamese-keyboard application, may be seen as a virus because of its hook functions, which are functionally similar to key logger threats. On the other hand, the second type of error is the false-negative. It refers to a failure identifying files containing harmful code but acting (or pretending) to be normal. Examples include malicious adware programs often integrated into free downloaded software [10]. In these cases, end-users, rather than anti-virus systems, may identify abnormal activities on their computers, e.g. too high memory use and large network traffic.

Addressing these errors, many studies [e.g. 11, 12, 13] propose techniques for analysing virus samples and deriving significant virus behaviour. These techniques allow building crucial knowledge bases over time [3]. Although these studies improved the overall effectiveness of the behaviour-based approach, they may not be comprehensive in term of its diversity and timeliness of analysed datasets. The main reason is that knowledge bases have to be shaped and therefore may take time to be updated. Besides, by only analysing a pre-gathered virus datasets, the deriving knowledge base is not thorough, because it does not consider behaviour of new used applications, which "are unknown and therefore have no expected normal behaviour" (p. 64) [1]. As a result, the two aforementioned problems still remain, especially the problem of detecting a new benign application as a virus.

When deploying the D32 (also known as D2) anti-virus project [14], we identified the problems caused by having incomplete knowledge base. Therefore, we suggest that end-users may play important roles in solving the problem. Users that develop knowledge about malicious behaviour by using applications can suggest whether applications are benign or malicious. Besides, collecting users' data about application behaviour may help building confidence in detecting and classifying viruses. However, the current literature does not propose any framework or technique considering the role of end-users in building anti-virus knowledge bases.

Since crowdsourcing was first introduced by Howe [15] as a strategy that relies on the crowd to achieve specific tasks, the crowdsourcing model has been suggested for doing tasks that require large human resources, like building knowledge bases [16]. Indeed, crowdsourcing has been utilized for building knowledge bases in different application domains. For instance, Wikipedia is a typical example of organizations successfully applying a crowdsourcing strategy when using a significant number of its anonymous users to perform writing and editing activities [17, 18]. Recently, Vukovic et al. [19] deployed a crowdsourcing application to capture IT Inventory knowledge. Other examples of building knowledge bases by the crowd have been reported by [20-22].

The current work in progress proposes a new architecture utilizing the crowd for building an anti-virus knowledge base. This architecture extends the research by Truong and Hoang [8] that introduced a machine learning algorithm for analysing virus datasets. In particular, we introduce mechanisms to collects users' feedback on application behaviour, and then use both internal and the crowd anti-virus experts for analysing application behaviour and classifying received feedbacks. A machine learning algorithm is applied to analyse these data, and results are used to update the anti-virus knowledge base. We call this architecture CrowdMAV (Applying **Crowd**sourcing to **M**achine Learning **A**nti-**V**irus System).

The main contribution of this work-in-progress is the CrowdMAV architecture that, for the first time, utilizes a crowdsourcing strategy for building a knowledge base of an anti-virus malicious and benign behaviour. This database can reduce the detection of false-positive. Another expected contribution of the current work is its mechanism to handle a huge amount of data receiving from the crowd by also utilizing anti-virus experts from the crowd, which is known as one mechanism of result aggregation in the crowdsourcing field [18]. From a practical perspective, this study helps improving the overall performance of the behaviour-based technique.

2 Related Work

2.1 Anti-virus Detection Techniques

A computer virus is defined as "a program that can 'infect' other programs by modifying them to include a possibly evolved copy of itself" (p. 23) [23]. Computer viruses can degrade the performance of a computer by disabling, damaging and destroying computer resources [8], gathering private data, and using resources in unintended ways. With the widespread of computer viruses and malwares, anti-virus software from different vendors has become very popular, in trying to prevent problems by identifying and stopping viruses immediately when they have entered a computer [1]. The existing anti-virus software generally deploys one of the two following detection techniques: a signature-based technique and the behaviour-based technique.

In the signature-based technique, the anti-virus software scans every received (or copied) file with code and compares them with known threat signatures [1]. In particular, this technique requires having an up-to-date signature database extracted from known threats [13], which is often updated daily by vendors. Within the database, each virus has a unique tag that is used to classify suspect files. When a computer receives a new file, the signature-based anti-virus software analyses the content of the received file to determine if it has known malicious tags. Although this technique was widely used in the past, it has two major problems. First, this technique cannot identify viruses that are not recorded in the knowledge base, including polymorphic viruses [1]. Second, this technique needs to analyse file code, which is difficult or sometimes impossible because of obfuscated or packed viruses [4].

Up to now, behaviour-based technique has been studied for more than a decade. This technique dynamically examines unknown files and monitors the file code execution in a controlled environment to detect its malicious behaviour [1]. Several types of behaviour are analysed. For instance, memory usage is examined in dynamic taint analysis [24]. Bayer et al. [4] suggest analysing execution traces. Other behaviour that is typically analysed is Windows API or system calls [3, 25], information flow [11] and network messaging [12]. Classifying the existing malicious behaviour, Hu [26] finds six classes of behaviour, including file-related, process-related, window-related, network-related, register-related, and windows-service behaviour.

Depending on the types of behaviour that are analysed, anti-virus vendors build their knowledge bases, "representing the execution behaviour of a family of malware instances" (p. 1) [3] and consisting of rules for detecting viruses [8]. These knowledge bases largely influence the effectiveness of anti-virus systems. Therefore, several efforts have been made to develop and improve the quality and completeness of knowledge bases [8, 27]. However, despite these efforts, the false-positive rates remain high [1]. We believe that one reason for this failure is that existing anti-virus knowledge bases do not consider new applications used by end-users [1], and thus the corresponding anti-virus system may classify the benign behaviour from these applications as malicious because the knowledge bases were not updated. Another problem with the current knowledge bases is that virus developers are well aware of the behavioural attitudes detected and developed counter measures, as stated that "we have to accept that virus authors are one step more ahead because they decide how to attack first" (p. 7) [2]. To address this problem, anti-virus software needs knowledge bases built from application behaviour reported by the world wide end-users. Addressing this need, we propose an architecture utilizing the crowd for building and extending the knowledge base.

2.2 Crowdsourcing for Building Anti-Virus Knowledge Bases

'Crowdsourcing' is a concept introduced in 2006 by Howe [15], who referred to crowdsourcing as a model utilizing the crowd for achieving organizational tasks. Since its introduction, crowdsourcing has been widely studied and conceptualized by several researchers, leading to the existence of different definitions. For instance, some authors compared crowdsourcing to the concept of outsourcing [15, 28]. Others considered crowdsourcing as a model for micro-tasks, where users provide their free time to accomplish particular tasks [18, 29, 30]. Recently, Estellés-Arolas and González-Ladrón-de-Guevara [31] analysed the existing definitions of crowdsourcing, and proposed an integrated definition. However, the definition is wordy and complex [32], thus we already adapted and simplified in into the following definition in our previous work [33], which is also used in the current study.

Crowdsourcing is defined as an online strategy, in which an organisation proposes defined task(s) to the members of the crowd via a flexible open call. By undertaking the task(s), the members contribute their work, knowledge, skills and/or experience, and receive reward. The organisation will obtain these contributions and utilize the results for the defined goals.

Literature has also showed that crowdsourcing can be utilized for different applications. Howe [15] discusses the concept of crowdsourcing through several real business applications, including iStockphoto for images exchange, InnoCentive for problem solving, and Amazon Mechanical Turk for micro tasks. Not only limited to business applications, crowdsourcing can also be applied in scientific research [34], urban planning [35], and cultural heritage [36]. Through the success of these initiatives, the literature recommends that crowdsourcing can leverage expertise, information, skills, and labour [37-39]. In particular, crowdsourcing is very helpful for tasks that need a large amount of human labour and cognitive abilities that are hard for computers to reproduce [30, 40].

Given the ability of crowdsourcing for achieving tasks that cannot be automated and need large workforce, using the crowd for building anti-virus knowledge bases seems a promising approach. Indeed several studies have reported their success on utilizing crowdsourcing for building knowledge bases in a variety of contexts [19, 20] [41]. To clarify more, here we introduce two interesting cases. The first case is the well-known encyclopaedia, Wikipedia, which is one of the most successful stories of using the crowd to perform tasks in the last decade. Since its introduction in 2001, this encyclopaedia has achieved 21 million users [42], who are both readers and contributors for contents of the encyclopaedia. It would certainly be impossible for Wikipedia to build one of the largest knowledge bases in human history without utilizing the ability of the crowd. The second case is reported by McCoy et al. [41], where crowdsourcing was applied to build a knowledge base in the medical area, which requires high knowledgeable and precise information. Utilizing the crowd for generating problem–medication pairs, these authors concluded that "crowdsourcing is an effective, inexpensive method for generating an accurate, up-to-date problem-medication knowledge base," (p. 5) [41].

In spite of these promising capabilities, only a few studies have considered crowdsourcing in the context of virus detection, where the crowd inputs can help overcoming high false-positive rates, as discussed in Section 2.1. One of these studies is the work by Burguera et al. [43]. Focusing on the Android platform, these authors proposed a framework collecting application data to detect malwares in Android devices. Utilizing data from end-users for detecting viruses, our framework however is different from the work by Burguera et al. [43] in two aspects. First, we are not limited to Android malware, but address viruses and malware in general. Second, the users' inputs in our architecture will be processed by internal and crowd anti-virus experts, and then a machine learning algorithm is applied for building the anti-virus knowledge base.

3 Utilizing the Crowd for Building an Anti-virus Knowledge Base

In this section, we propose an architecture utilizing the crowd for building an anti-virus knowledge base. As the architecture utilizes crowdsourcing to extend the Machine Learning Approach to Anti-virus System [8], we named it CrowdMAV. This section starts by overviewing on the business process crowdsourcing of CrowdMAV, which involves three main activities (Fig. 1).

First, *an open-call* is delivered to the internet users, asking them for providing inputs to the CrowdMAV. The call is delivered to the crowd through two channels. It is posted in the official D32 anti-virus website [14], along with a delivery of a trial version of the D32 anti-virus software (D32). For existing D32 users, D32 shows a message asking their participation in the crowdsourcing process. Currently, we provide a trial (or extended) period of using D32 as an incentive for user participation.

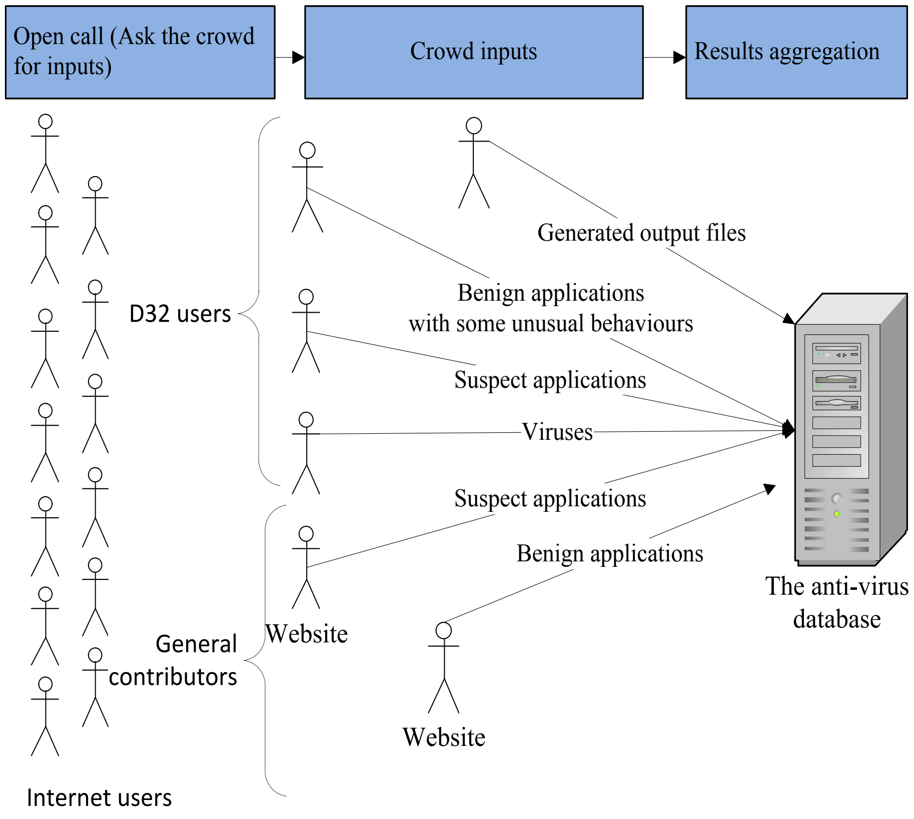


Fig. 1. Business process crowdsourcing of CrowdMAV

Second, when participating in the CrowdMAV, D32 users or general contributors *submit their inputs*. For D32 users, the anti-virus software automatically asks users to confirm the suspect behaviour. Alternatively, D32 users can also manually raise suspect application behaviour and send their inputs when they found virus-related problems, such as benign applications that were detected as viruses, slow response, or unauthorised access to a website. In these cases, these users can activate a form within D32, which then guides their submission. Other general contributors, who may use anti-virus software different from D32, can also submit their inputs through the CrowdMAV website or send emails. We note that although the inputs can be submitted through different channels, they should include the following information: attached file(s), whether the files should be seen as virus or benign applications that were wrongly detected as a virus, the suspect behaviour, e.g. unauthorized access to a website or continuously reading the hard disk, and user messages that provide more information for the submission. For those who use D32, we additionally collect application behaviour-related data, but personal data will not be collected. We note that each user over time can provide more than one submission to the CrowdMAV.

The final activity is related to *results aggregation.* The users' inputs coming from different channels are sent to the anti-virus server. Receiving these inputs, the server processes each one as a dataset. Thus, the larger numbers of users participate in the CrowdMAV, the more datasets can be collected and processed by the server, leading to more thorough anti-virus knowledge base by the end. Aligning with the classification by [44], the server classifies the datasets into three groups: suspect, benign, and unclassified objects. This classification is based on the users' suggestions included within the inputs. If the users' inputs are consistent on whether an application is suspect or benign, this application is classified to the corresponding group. If there are conflicted user opinions on certain datasets, e.g. new software that is suggested as a malicious application by some users but seen as a benign application by other users, the datasets are grouped as unclassified objects. All groups of datasets are then updated into the database, and the CrowdMAV will use these datasets to develop the anti-virus knowledge base, as presented in the following architecture.

3.1 The architecture of CrowdMAV

The results from the previous process are aggregated by CrowdMAV. Fig. 2 shows the architecture of the CrowdMAV, which relies on three components. The first component is *virus detection*, which is responsible for analysing the datasets received by the anti-virus server to detect viruses and malware. The virus detection is processed by the Virus Scanning Agent (VSA) that is also embedded in the D32 anti-virus software (D32) [8]. One can argue that the users have already used D32 to scan these files in their own computers before sending them to the server, and therefore rescan is unnecessary. The reason for doing it again in the architecture is that the users' D32 installation is not always up-to-date, while the VSA in CrowdMAV uses the latest anti-virus knowledge base. As a result, the VSA can automatically classify some files without further processing. In particular, the detected and classified objects are checked against the anti-virus knowledge base. Regarding the three aforementioned groups of datasets, we believe that the VSA is effective in processing the suspect group because it is regularly updated with new virus samples collected from different sources, e.g. http://openmalware.org. This component can also classify some files in the unclassified object group. However, this component faces difficulties when classifying the last group that were suggested as malicious applications by the anti-virus software but reported as benign files by users.

The objects that cannot be classified by the virus detection are analysed by *expert evaluation* (Component 2). As a crowdsourcing approach often returns a huge amount of data [22], i.e. unclassified objects in our case, we combined our available experts with the experts in the crowd for evaluation. The decision on who will be considered as experts and invited to participant in the evaluation is made by the D2 company. After invited, these experts are communicated through a web application, similar to the work by [19]. This application then chooses the most frequent unclassified objects received from the users to ask for expert evaluation. It then provides a controlled environment for each expert to execute unclassified objects. Using some heuristics [45], the expert makes his decision on the classified objects. In case the decisions are consensus, it is the final decision on object classification. In case of conflicted decisions, i.e. malicious or benign, the decision supported by the majority of experts will be chosen as the final decision. The results are updated into the knowledge base. We note that the results of this component are not only classified objects, but also new suggested rules and application behaviour for detection and classification.

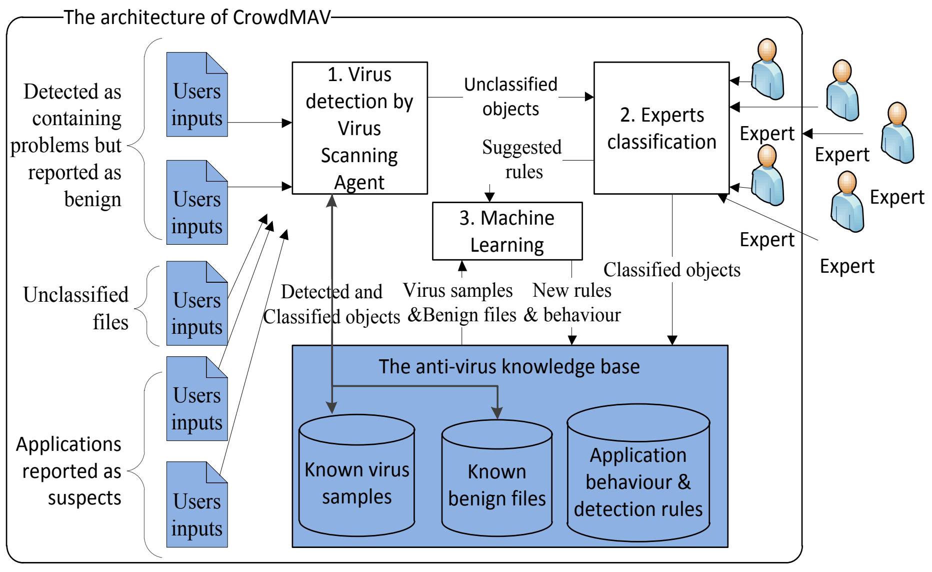
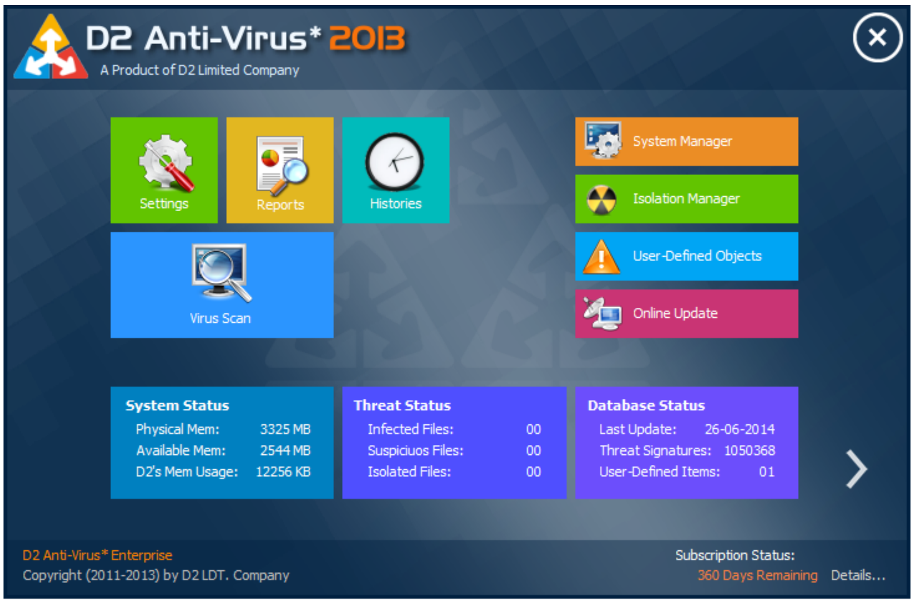

The architecture of CrowdMAV
Detected as containing problems but reported as benign
Unclassified files
Applications reported as suspects
Users inputs
Users inputs
Users inputs
Users inputs
Users inputs
1. Virus detection by Virus Scanning Agent
Unclassified objects
Suggested rules
2. Experts classification
3. Machine Learning
Expert
Expert
Expert
Expert
Expert
Detected and Classified objects
Virus samples &Benign files
New rules & behaviour
Classified objects
The anti-virus knowledge base
Known virus samples
Known benign files
Application behaviour & detection rules

Fig. 2. The architecture of the CrowdMAV for building the anti-virus knowledge base

As a result of these two components, new objects are classified in the knowledge base. Using the updated knowledge base, the *machine learning component* (component 3) is responsible for discovering new detection rules. Extending from the work by Truong and Hoang [8], the machine learning is firstly trained using a list of known virus samples and benign files. As the list evolves, the component updates its detection rules based on the rules suggested by both internal and crowd experts. Finally, the machine learning component will be retrained over the updated knowledge base. As a result, new application behaviour and detection rules are identified and added to the anti-virus knowledge base.

4 Implementation (Work-in-progress)

A part of this work has already been implemented. In particular, we have already developed the virus detection as an application and collect a number of virus sample that will be used for training the machine learning (Virus Scan function and Database Status in Fig. 3).



System Status
Physical Mem: 3325 MB
Available Mem: 2544 MB
D2's Mem Usage: 12256 KB

Threat Status
Infected Files: 00
Suspiciuos Files: 00
Isolated Files: 00

Database Status
Last Update: 26-06-2014
Threat Signatures: 1050368
User-Defined Items: 01

Subscription Status:
360 Days Remaining

Fig. 3. Virus detection component and known virus database

Within D32, users can access the 'reports' function to call the submission form (Reports function in Fig. 3). The detailed submission form is presented in Fig. 4. For internet users, they can use our website for submitting suspect applications [14]. However, the form and website for collecting data need to be extended as currently they are focusing on collecting suspect files. In particular, the extension should focus on gathering data that meet the new structure of the anti-virus knowledge base, and thus should include the following data: suspect files, whether the files are virus or benign, their suspect behaviour, and messages from the users.

We also deployed the anti-virus server in order to collect users input from D32 and the website [14]. We are implementing the process required to aggregate users' submissions. In the expert evaluation component, although we already have a team of internal experts, we need to engage more experts from the crowd for detecting viruses and classifying suspect objects. Consequently, a web application for these experts to communicate and process their classification activities is needed. Another activity that is receiving our focus is the extension of the machine learning algorithm proposed by [8] to improve its ability of learning, corresponding to the new knowledge base structures. Additionally, some parts of the current user interface in D32 website are presented in Vietnamese, and need to be translated into English.

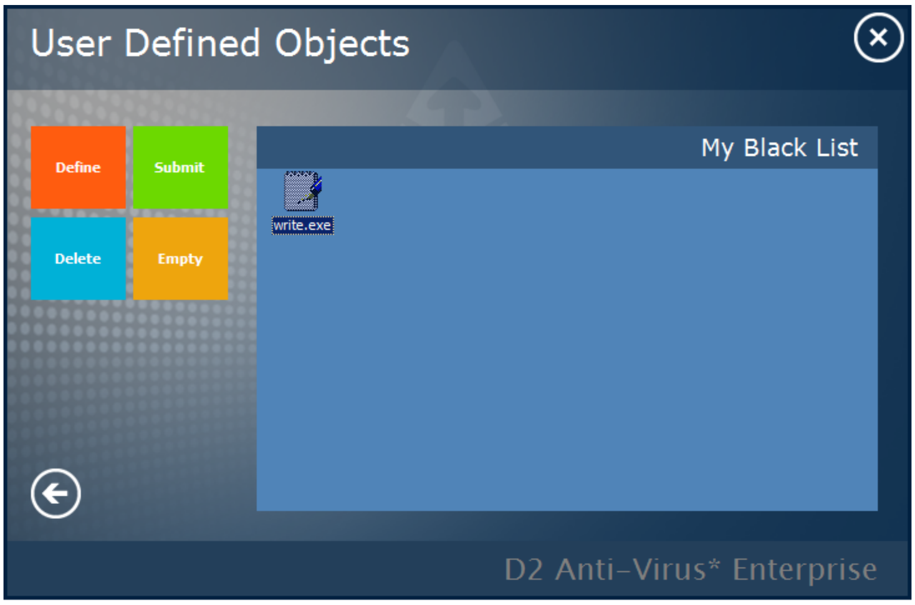


Fig. 4. A form for submitting users' inputs

5 Conclusion and Future Work

We proposed CrowdMAV, an architecture for updating the anti-virus knowledge base utilizing the crowdsourcing strategy. Addressing the high false-positive rates found in the behaviour-based techniques [1], our architecture utilized users' inputs in order to extend the anti-virus knowledge base. The users' inputs are first classified by the VSA and then by expert evaluation, in which we also suggested a crowdsourcing approach by combining internal and crowd experts. The classified data are mined by the machine learning algorithm [8], resulting in new rules and new objects classified in the knowledge base. By doing so, the knowledge base includes new information learned from the crowd, and as a result, the system can correctly diagnose and classify objects that could not be classified previously.

In comparison to other studies [3, 4, 11, 25], our work has a similar aim to improve the effectiveness of behaviour-base detection techniques. However, we chose a different focus. While other works focus on analysing application behaviour to detect benign and malicious applications [3, 4, 11, 25], our architecture aims at building a comprehensive knowledge base, necessary for the analysis. Although this focus has received attention from several anti-virus vendors, such as Norton anti-virus from Symantec [46] and Malware Protection Centre from Microsoft [47], only a few studies examined the use of crowdsourcing to improve anti-virus knowledge base (e.g. [43]). Our architecture fulfils this gap by proposing an architecture considering the role of crowdsourcing for building the anti-virus knowledge base.

From a practical point of view, our framework, by enriching the anti-virus knowledge base, can improve the performance of the behaviour-based technique. To an extent, although our architecture is targeted to the D32 anti-virus software, we believe that this approach can also be applied to other anti-virus systems. It can also be combined with other detection techniques, i.e. signature-based techniques. As seen via Fig. 2, our architecture also updates the know virus database, which can enrich the virus sample for the signature-based techniques.

Given that this is a work in progress, we acknowledge that future studies are needed to solidify the architecture. In particular, the future work should be seen from two perspectives: crowdsourcing and anti-virus systems. From crowdsourcing point of view, the business process crowdsourcing in the current study addressed three activities (Fig. 1). We understand that business process crowdsourcing involves other aspects, such as incentive mechanism, crowd management, and workflow design [48], which need to be considered in further development of our architecture. From an anti-virus systems' view, we need to complete the machine learning algorithm with the new knowledge base and test the effectiveness of the architecture. This effectiveness should base on not only the number of correctly recognized viruses (true positive and true negative rates) but also false-positive and false negative rates, which reflect the current limitation of the behaviour-based techniques [1].

References

1. Sukwong, O., H.S. Kim, and J.C. Hoe, *Commercial antivirus software effectiveness: an empirical study*. Computer, 2011. **44**(3): p. 0063-70.
2. Rad, B.B., M. Masrom, and S. Ibrahim, *Evolution of computer virus concealment and anti-virus techniques: a short survey*. arXiv preprint arXiv:1104.1070, 2011.
3. Park, Y., D.S. Reeves, and M. Stamp, *Deriving common malware behavior through graph clustering*. Computers & Security, 2013. **39**: p. 419-430.
4. Bayer, U., et al. *Scalable, Behavior-Based Malware Clustering*. in *NDSS*. 2009. Citeseer.
5. Egele, M., et al., *A survey on automated dynamic malware-analysis techniques and tools*. ACM Computing Surveys (CSUR), 2012. **44**(2): p. 6.
6. Lin, D. and M. Stamp, *Hunting for undetectable metamorphic viruses*. Journal in computer virology, 2011. **7**(3): p. 201-214.

7. Hu, X., T.-c. Chiueh, and K.G. Shin, *Large-scale malware indexing using function-call graphs*, in *Proceedings of the 16th ACM conference on Computer and communications security*. 2009, ACM: Chicago, Illinois, USA. p. 611-620.
8. Truong, M.N.Q. and T.N. Hoang, *A multi-agent mechanism in machine learning approach to anti-virus system*, in *Agent and Multi-Agent Systems: Technologies and Applications*. 2008, Springer Berlin Heidelberg. p. 743-752.
9. Rieck, K., et al., *Learning and Classification of Malware Behavior*, in *Detection of Intrusions and Malware, and Vulnerability Assessment*, D. Zamboni, Editor. 2008, Springer Berlin Heidelberg. p. 108-125.
10. Microsoft. *Evolution of Malware*. 2014; Available from: http://www.microsoft.com/security/sir/story/default.aspx#!10year_malware.
11. Yin, H., et al. *Panorama: capturing system-wide information flow for malware detection and analysis*. in *Proceedings of the 14th ACM conference on Computer and communications security*. 2007. ACM.
12. Stinson, E. and J. Mitchell, *Characterizing Bots' Remote Control Behavior*, in *Detection of Intrusions and Malware, and Vulnerability Assessment*, B. M. Hämmerli and R. Sommer, Editors. 2007, Springer Berlin Heidelberg. p. 89-108.
13. Schultz, M.G., et al. *Data mining methods for detection of new malicious executables*. in *Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on*. 2001.
14. D32. *D32 Anti-virus*. 2014; Available from: http://www.d32av.vn/.
15. Howe, J., *The rise of crowdsourcing*, in *Wired Magazine 2006*. 2006, Dorsey Press. p. 1-4.
16. Muntés-Mulero, V., et al., *Crowdsourcing for industrial problems*, in *Citizen in Sensor Networks*, J. Nin and D. Villatoro, Editors. 2013, Springer Berlin Heidelberg. p. 6-18.
17. Chi, E.H. and M.S. Bernstein, *Leveraging Online Populations for Crowdsourcing.* Internet Computing, IEEE, 2012. **16**(5): p. 10-12.
18. Zhao, Y. and Q. Zhu, *Evaluation on crowdsourcing research: Current status and future direction.* Information Systems Frontiers, 2012: p. 1-18.
19. Vukovic, M., J. Laredo, and S. Rajagopal, *Challenges and experiences in deploying enterprise crowdsourcing service*, in *Web Engineering*, B. Benatallah, et al., Editors. 2010, Springer Berlin Heidelberg. p. 460-467.
20. Fraternali, P., et al., *Putting humans in the loop: Social computing for Water Resources Management.* Environmental Modelling & Software, 2012. **37**(0): p. 68-77.
21. Corney, J., et al., *Putting the crowd to work in a knowledge-based factory.* Advanced Engineering Informatics, 2010. **24**(3): p. 243-250.
22. Doan, A., R. Ramakrishnan, and A.Y. Halevy, *Crowdsourcing systems on the world-wide web.* Communications of the ACM, 2011. **54**(4): p. 86-96.
23. Cohen, F., *Computer viruses: theory and experiments.* Computers & security, 1987. **6**(1): p. 22-35.
24. Clause, J., W. Li, and A. Orso. *Dytan: a generic dynamic taint analysis framework*. in *Proceedings of the 2007 international symposium on Software testing and analysis*. 2007. ACM.
25. Willems, C., T. Holz, and F. Freiling, *Toward automated dynamic malware analysis using cwsandbox.* IEEE Security and Privacy, 2007. **5**(2): p. 32-39.
26. Hu, Y., et al. *Unknown malicious executables detection based on run-time behavior*. in *Fuzzy Systems and Knowledge Discovery, 2008. FSKD'08. Fifth International Conference on*. 2008. IEEE.
27. Lanzi, A., M.I. Sharif, and W. Lee. *K-Tracer: A System for Extracting Kernel Malware Behavior*. in *NDSS*. 2009.

28. Rouse, A.C., *A preliminary taxonomy of crowdsourcing.* Proceedings of the 21st Australasian Conference on Information Systems, 2010: p. 1-10.
29. Kittur, A., E.H. Chi, and B. Suh. *Crowdsourcing user studies with Mechanical Turk.* in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2008. ACM.
30. Sarasua, C., E. Simperl, and N.F. Noy, *Crowdmap: Crowdsourcing ontology alignment with microtasks*, in *The Semantic Web–ISWC 2012*. 2012, Springer. p. 525-541.
31. Estellés-Arolas, E. and F. González-Ladrón-de-Guevara, *Towards an integrated crowdsourcing definition.* Journal of Information science, 2012. **38**(2): p. 189-200.
32. Brabham, D.C., *Crowdsourcing*. 2013, Canbridge, MA: The MIT Press.
33. Thuan, N.H., P. Antunes, and D. Johnstone, *Factors Influencing the Decision to Crowdsource*, in *Collaboration and Technology*, P. Antunes, et al., Editors. 2013, Springer Berlin Heidelberg. p. 110-125.
34. Mason, W. and S. Suri, *Conducting behavioral research on Amazon's Mechanical Turk.* Behavior research methods, 2012. **44**(1): p. 1-23.
35. Brabham, D.C., *Motivations for Participation in a Crowdsourcing Application to Improve Public Engagement in Transit Planning.* Journal of Applied Communication Research, 2012. **40**(3): p. 307-328.
36. Kingston, A., *"Choir attempted that beautiful anthem "Oh, Radiant Morn" – made a hash of it" - Making a hash of the Adkin Diary transcriptions*, in *Workshop on Crowdsourcing for the Digital Humanities and Cultural Heritage Sector*. 2013: Wellington, New Zealand.
37. Brabham, D.C., *Crowdsourcing as a Model for Problem Solving: An Introduction and Cases.* Convergence: The International Journal of Research into New Media Technologies, 2008. **14**(1): p. 75-90.
38. Vukovic, M. and C. Bartolini, *Towards a research agenda for enterprise crowdsourcing*, in *Leveraging applications of formal methods, verification, and validation*, T. Margaria, Steffen, B., Editor. 2010, Springer Berlin Heidelberg. p. 425-434.
39. Aitamurto, T., A. Leiponen, and R. Tee, *The Promise of Idea Crowdsourcing–Benefits, Contexts, Limitations*, in *White Paper for Nokia IdeasProject. June*. 2011.
40. Franklin, M.J., et al., *CrowdDB: answering queries with crowdsourcing*, in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. 2011, ACM: Athens, Greece. p. 61-72.
41. McCoy, A.B., et al., *Development and evaluation of a crowdsourcing methodology for knowledge base construction: identifying relationships between clinical problems and medications.* Journal of the American Medical Informatics Association, 2012. **19**(5): p. 713-718.
42. Wikipedia. *Statistics.* 2014 [cited 2014 June 2014]; Available from: http://en.wikipedia.org/wiki/Special:Statistics.
43. Burguera, I., U. Zurutuza, and S. Nadjm-Tehrani. *Crowdroid: behavior-based malware detection system for android.* in *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*. 2011. ACM.
44. Saeed, I.A., et al., *A Survey on Malware and Malware Detection Systems.* analysis, 2013. **3**(10): p. 13-17.
45. Adkins, F., et al. *Heuristic malware detection via basic block comparison*. in *Malicious and Unwanted Software:" The Americas"(MALWARE), 2013 8th International Conference on*. 2013. IEEE.
46. Symantec. *Submit Virus Samples*. 2014 [June 2014]; Available from: http://www.symantec.com/security_response/submitsamples.jsp.

47. Microsoft. *Submit a sample*. 2014 [June 2014]; Available from: https://www.microsoft.com/security/portal/submission/submit.aspx.
48. Thuan, N.H., P. Antunes, and D. Johnstone, *Toward a Nexus Model Supporting the Establishment of Business Process Crowdsourcing*, in *1st International Conference on Future Data and Security Engineering (FDSE 2014)*, T.K. Dang, et al., Editors. 2014, Springer Berlin Heidelberg.