Supporting Knowledge-assisted Rule Creation in a Behavior-based Malware Analysis Prototype

Johannes Schick[1], Markus Wagner[2], Niklas Thür[2], Christina Niederer[2], Gernot Rottermanner[2],
Paul Tavolato[2], Wolfgang Aigner[2]
University of Applied Sciences, St. Pölten, Austria
Email: [1]dm171566@fhstp.ac.at, [2]first.last@fhstp.ac.at

Abstract

The ever increasing number of malicious software (malware) requires domain experts to shift their analysis process towards more individualized approaches to acquire more information about presently unknown malware samples. KAMAS is a knowledge-assisted visual analytics prototype for behavioral malware analysis, which allows IT-security experts to categorize and store potentially harmful system call sequences (rules) in a knowledge database. In order to meet the increasing demand for individualization of analysis processes, analysts have to be able to create individual rules. This paper is a visualization design study, which describes the design and implementation of a separate Rule Creation Area (RCA) into KAMAS and its evaluation by domain experts.It became clear that continuous integration of experts in interaction processes improves the analysis and knowledge generation mechanism of KAMAS. Additionally, the outcome of the evaluation revealed that there is a demand for adjustment and re-usage of already stored rules in the RCA.

Index Terms: K.6.1 [Information Interfaces and Presentation]: User Interfaces—User-centered design—Evaluation/methodology;

1 Introduction & Related Work

Nowadays, domain experts have to deal with an ever increasing number of malicious software (malware) which in addition is becoming more targeted, persistent and unknown [4]. In behavior-based analysis, malware analysts have to deal with large amounts of data, which can lead to a very complicated analyzing process: a trace of a malware sample may often comprise thousands of system calls and analysts have to find similar system call patterns within thousands of such traces. In order to simplify this process, analysts need automated approaches for finding such patterns and categorizing them as potentially harmful or harmless. However, such identification of patterns relies heavily on the analysts knowledge, which makes it impossible to automate this process completely [8]. These patterns of behaviors can be defined as a formal language using formal grammars (syntactic pattern recognition [3], [5]). See [2] for details. The task of the analyst is the development of a set of grammar rules incorporating his/her knowledge about (malicious) behaviors of malware samples. In this context, visual analytics (VA) is needed to support the analysts in integrating their knowledge.

According to Keim et al. [6], VA also connects automated analysis techniques with interactive visualizations, combining different types of information and obtain understanding from complex data sets. To make reasoning out of this massive amount of data, it is necessary to include ”implicit” [1] or ”tacit” [11] knowledge in the analysis process. By externalizing the implicit/tacit knowledge of domain experts, it is possible to provide explicit knowledge in form of data, which is independent from the current user of the system. This extracted knowledge can subsequently be connected through interactive visualization tools [11]. Since there were no VA tools available covering all requirements for malware analysts, Wagner et al. [10] developed a Knowledge-Assisted Visual Malware Analysis System (KAMAS). With KAMAS, analysts are able to categorize function call traces in terms of their potential harmfulness and store them into a knowledge database (KDB). To improve the effectiveness of KAMAS, Wagner et al. [9] suggested an interface design for a separate Rule Creation Area (RCA), which allows the construction and storage of new rules in the KDB.

In general, this paper contributes as a design study [7], following a problem-oriented research approach. This includes a problem definition, the design and implementation of a visualization system to solve the problem, the evaluation of the prototype as well as a reflection [7]. The problem was defined by Wagner et al. [9] in their design study, which addresses the need for the implementation of a separate area for rule creation in the KAMAS prototype.

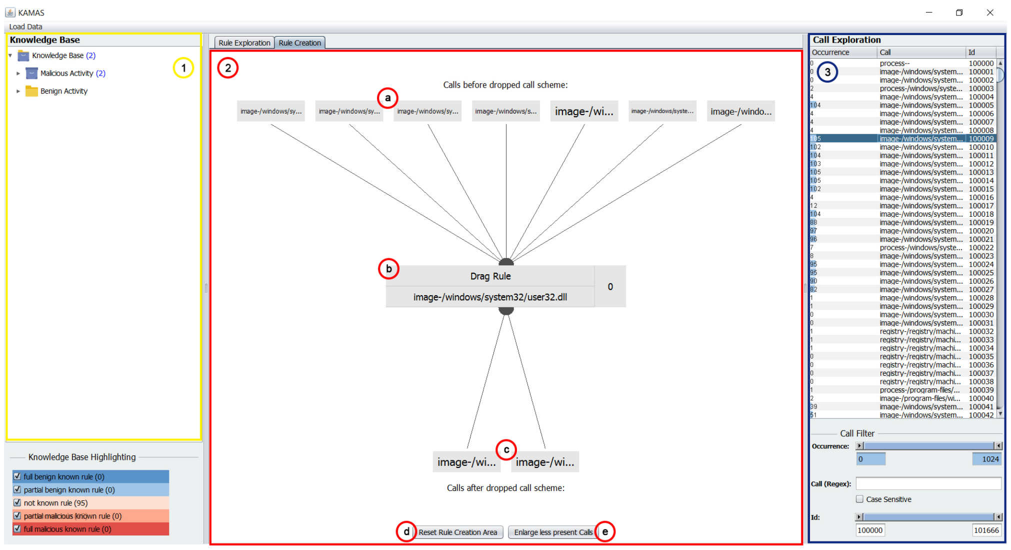
2 Rule Creation Area Concept

RCA in General: The Rule Creation Area (RCA) consists of three main areas (see Figure 1.2). First, the analyst can drop single calls, which she previously selected and dragged from the ’Call Exploration’ table into the Rule Creation Table (RCT) (see Figure 1.2.b). Secondly, above and below the RCT, the interface provides suggestions for single calls which occur either before (see Figure 1.2.a) or after (see Figure 1.2.c) the dropped system call sequence. At last, on the bottom of the RCA the analyst has the possibility to reset the whole RCA to its default state (see Figure 1.2.d) and to switch between the highlighting of more or less frequent call suggestions (see Figure 1.2.e).

Rule Creation in the RCA: The top-row of the RCT makes it possible to drag and drop a newly created rule into the KDB. Furthermore, the number in the second column of the RCT represents the occurrence of the newly created rule in the loaded file. By dragging a single call and move it to another position, a reordering of calls inside the created rule can be performed. Also, single calls can be deleted from the RCT by right clicking on the desired call and using a ’Delete’ pop-up. Both actions affect the occurrence column and the automatically provided call suggestions (supporting the rule creation) in the RCA. The suggestions above (see Figure 1.2.a) and below (see Figure 1.2.c) the RCT are validated depending on the currently dropped system call sequence and the loaded file. The calls above occur before the dropped sequence whereas the calls below occur afterwards. Moreover, the font size of the call suggestions varies depending on their occurrence. Thus, the analyst gets a better overview of which single calls are more or less frequent.

3 Evaluation, Reflection & Conclusion

During the performed design study [7], we evaluated the newly implemented functionalities with real world users. A case study with two malware analysis experts was conducted. Next, we summarize the most important findings and reflect on them:



Occurrence	Call	Id
0	process--	100000
0	image-/windows/system...	100001
0	image-/windows/system...	100002
2	process-/windows/syste...	100003
4	image-/windows/system...	100004
104	image-/windows/system...	100005
4	image-/windows/system...	100006
4	image-/windows/system...	100007
4	image-/windows/system...	100008
105	image-/windows/system...	100009
102	image-/windows/system...	100010
104	image-/windows/system...	100011
103	image-/windows/system...	100012
105	image-/windows/system...	100013
105	image-/windows/system...	100014
102	image-/windows/system...	100015
4	image-/windows/system...	100016
12	image-/windows/system...	100017
104	image-/windows/system...	100018
88	image-/windows/system...	100019
97	image-/windows/system...	100020
96	image-/windows/system...	100021
7	process-/windows/syste...	100022
8	image-/windows/system...	100023
95	image-/windows/system...	100024
95	image-/windows/system...	100025
90	image-/windows/system...	100026
82	image-/windows/system...	100027
1	image-/windows/system...	100028
1	image-/windows/system...	100029
0	image-/windows/system...	100030
0	image-/windows/system...	100031
1	registry-/registry/machi...	100032
1	registry-/registry/machi...	100033
1	registry-/registry/machi...	100034
0	registry-/registry/machi...	100035
0	registry-/registry/machi...	100036
0	registry-/registry/machi...	100037
0	registry-/registry/machi...	100038
1	process-/program-files/...	100039
2	image-/program-files/wi...	100040
39	image-/windows/system...	100041
51	image-/windows/system...	100042

Figure 1: User interface of the KAMAS prototype with activated Rule Creation Area (RCA). 1) Knowledge Database (KDB) for storing newly created rules. 2) RCA including call suggestions before (2.a) and after (2.c) the currently dropped call sequence, the rule creation table (2.b), the button to reset the whole RCA (2.d) and the button to change the call suggestion size according to their occurrence (2.e). 3) 'Call Exploration' table with a list of all single calls included in the currently loaded file.

Consistency: To support the experts' needs [8], drag and drop operations served as the major interaction technique in this prototype. This involves the addition of single calls and call suggestions to the RCT, their reordering as well as storing of created rules in the KDB. Both analysts were comfortable with the handling of the given interaction possibilities. However, the evaluation showed that an additional visualization is needed to make the outcome of drag and drop operations fully transparent.

Creation Support: The implemented prototype provides call suggestions to accelerate and simplify the rule creation process. Based on the currently dropped call sequence, the system validates the call suggestions automatically. Additionally, the prototype offers a possibility to highlight more or less frequent call suggestions.

Editing Options: The prototype provides possibilities to delete and reorder single calls in the RCT as well as to restart the whole process from scratch. Both experts expressed their wish for reusing/adjusting already stored rules in the RCA. This feature can be considered as highly recommended to implement.

Knowledge Generation: The possibility to drag the created rule and store it in the KDB was also well received by the experts. The implementation of this feature expands the system's knowledge generation loop with more flexibility by integrating individually created rules. As a result, the overall knowledge generation process is getting more individualized and the following analysis process can draw upon different expertises.

Future Work: The usage of already stored rules for rule creation can be seen as the next logical step for further development of the presented prototype. Additionally, the enhancement of interaction visualization should round off the overall appearance and usability of the user interface.

Acknowledgments

This work was supported by the Austrian Science Fund (FWF) via the "KAVA-Time" project (P25489-N23). We would also like to thank all focus group members and test participants who have agreed to volunteer in this project.

References

[1] M. Chen, D. Ebert, H. Hagen, R. S. Laramee, R. v. Liere, K. L. Ma, W. Ribarsky, G. Scheuermann, and D. Silver. Data, Information, and Knowledge in Visualization. *IEEE Computer Graphics and Applications*, 29(1):12–19, 2009. doi: 10.1109/MCG.2009.6

[2] H. Dornhackl, K. Kadletz, R. Luh, and P. Tavolato. Malicious Behavior Patterns. In *IEEE International Symposium on Service Oriented System Engineering*, pp. 384–389, 2014. doi: 10.1109/SOSE.2014.52

[3] K. Fu. *Syntactic pattern recognition and applications*. Prentice-Hall advanced reference series: Computer science. Prentice-Hall, 1982.

[4] E. Gandotra, D. Bansal, and S. Sofat. Malware Analysis and Classification: A Survey. *Journal of Information Security*, 05(02):56, 2014. doi: 10.4236/jis.2014.52006

[5] R. Gonzalez and M. Thomason. *Syntactic pattern recognition: an introduction*. Addison-Wesley, 1978.

[6] D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann. *Mastering the Information Age Solving Problems with Visual Analytics*. Eurographics Association, 2010.

[7] M. Sedlmair, M. Meyer, and T. Munzner. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440, 2012. doi: 10.1109/TVCG.2012.213

[8] M. Wagner, W. Aigner, A. Rind, H. Dornhackl, K. Kadletz, R. Luh, and P. Tavolato. Problem Characterization and Abstraction for Visual Analytics in Behavior-based Malware Pattern Analysis. In *Proceedings of the Eleventh Workshop on Visualization for Cyber Security*, pp. 9–16. ACM, 2014. doi: 10.1145/2671491.2671498

[9] M. Wagner, A. Rind, G. Rottermanner, C. Niederer, and W. Aigner. Knowledge-assisted rule building for malware analysis. In *Proceedings of the 10th Forschungsforum der österreichischen Fachhochschulen*. FH des BFI Wien, FH des BFI Wien, Vienna, Austria, 2016.

[10] M. Wagner, A. Rind, N. Thür, and W. Aigner. A knowledge-assisted visual malware analysis system: Design, validation, and reflection of kamas. *Computers & Security*, 67:1–15, 2017. doi: 10.1016/j.cose.2017.02.003

[11] X. Wang, D. H. Jeong, W. Dou, S.-W. Lee, W. Ribarsky, and R. Chang. Defining and applying knowledge conversion processes to a visual analytics system. *Computers & Graphics*, 33(5):616–623, 2009. doi: 10.1016/j.cag.2009.06.004