Markus Wagner / Alexander Rind / Gernot Rottermanner / Christina Niederer / Wolfgang Aigner

Knowledge-Assisted Rule Building for Malware Analysis

103 - Recent Advances in Multimedia Processing, Organization and Visualization beyond Domains and Disciplines

Abstract

Due to the increasing threat from malicious software (malware), the monitoring of vulnerable systems is becoming increasingly important, which includes the need to log and analyze activity ranging from networks, individual computers, to mobile devices. Currently available tools in behavior-based malware analysis do not meet all experts' needs, such as selecting different rules, categorizing them by their task and storing them in the database as well as manually adapting and/or tuning the rules identified. To close this gap, we designed CallNet, a knowledge-assisted visual analytics and rule-building tool for behavior-based malware analysis. The paper at hand is a design study which describes the design, a usage scenario, and the paper prototype evaluation. We report on the validation of CallNet by expert reviews, reflect on the insights gained from the reviews and, finally discuss the advantages and disadvantages of the prototype design including the visualization techniques applied.

Keywords:

Visual analytics, interaction, behavior-based malware analysis, knowledge-assisted, knowledge generation, externalized knowledge, prior knowledge, user-centered design, prototyping, usability

1. Introduction & Related Work

An increasing number of malicious software (malware) samples are used for espionage and attacks against infrastructure. Thus, monitoring of vulnerable systems is becoming more and more important (e.g., Trinius et al. 2009; Yee et al. 2012; Dornhackl et al. 2014). Various detection approaches exist based on signature, behavior, or heuristics (Bazrafshan et al. 2013). Malicious software behavior is identified through static or dynamic analysis (Egele et al. 2008). When statically analyzing a possible malware sample, the binary file is usually disassembled and dissected function by function. In contrast, dynamic analysis observes a sample's runtime behavior (Egele et al. 2008). This is where Visual Analytics (VA) comes in, "the science of analytical reasoning facilitated by interactive visual interfaces" (Thomas & Cook 2005). A major tenet of VA is that analytical reasoning is not a routine activity that can be automated completely (Wegner 1997). Instead, it depends on analysts' initiative and domain experience. By using VA, it is possible to recognize patterns in the malware's code or behavior, and assist analysts in gaining better insights (Wagner et al. 2014). Existing literature explores this area from different points of view: Lee et al. (2011) concluded that it is necessary to use visualization for malware detection to recognize and extract unseen malware patterns. Shiravi et al.

(2012) present a survey of 38 network security visualization systems, but only some support methods for interactive data exploration. Conti (2007) dedicated a part of his book to visual representation of malware analysis with a focus on the network level. Additionally, Wagner et al. (2015) presented a survey on visualization tools for malware analysis, describing 25 visualization systems and categorizing them according to a new malware visualization taxonomy.

To be effective, VA needs to provide 'precise' data, "which is immediate, relevant and understandable to individual users, groups, or communities of interest" (Kielman et al. 2009). By externalizing this knowledge and using it, analysts can avoid cognitive overload and use visualization and automated analysis methods more effectively. Leading visualization researchers have repeatedly called for the integration of knowledge with visualization: Chen (2005) lists 'prior knowledge' as one of ten unsolved Information Visualization problems. Pike et al. (2009) point out that VA tools have only underdeveloped abilities to represent and reason with human knowledge. Therefore, they declare 'knowledge-based interfaces' as one of seven research challenges for upcoming years. This work presents CallNet, a new knowledge-assisted VA prototype that is based on a study to characterize and abstract behavior-based malware analysis as a VA problem (Wagner et al. 2014).

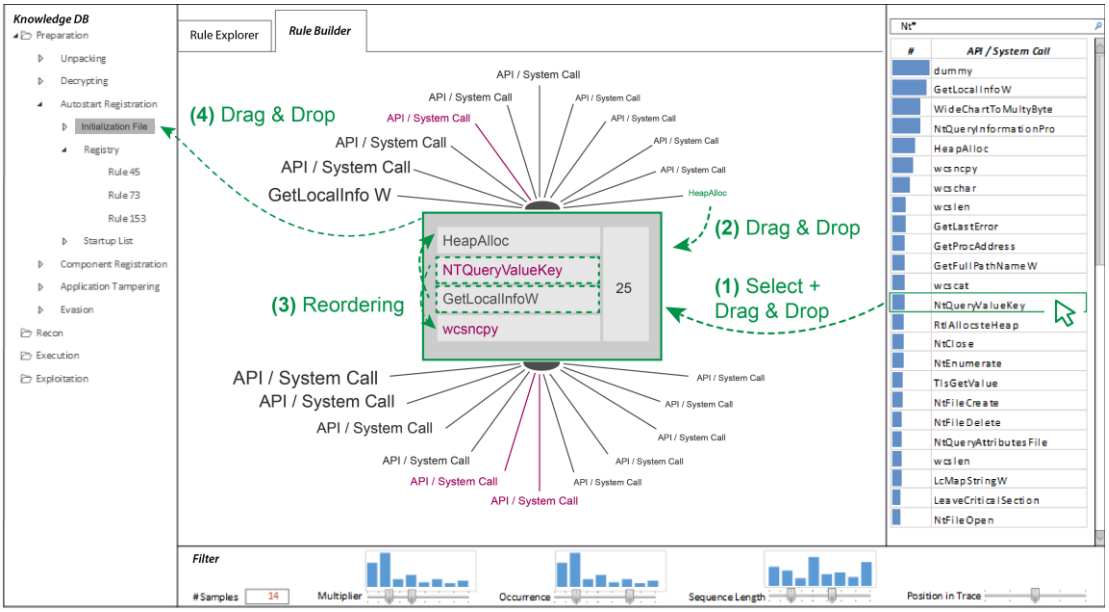

Figure 1: Interface of the CallNet prototype including the Knowledge Database (left), the Rule Building area (center), the Call Exploration area (right) and the different filtering abilities (bottom). Additionally, green overlays describe the workflow of the usage scenario (cf. Section 2.2).

2. Design & Evaluation

The workflow of malware analysis experts includes the exploration of malware execution traces in the form of system call sequences and frequently occurring subsequences. Their workflow generally involves the following tasks: "to select different rules, categorize them by their task and store them in the database as well as manual adaption and/or tuning of found rules". The prototype described in this paper allows analysts to generate system call sequences (rules) from scratch and evaluate them with respect to how often they occur in malware execution traces (Figure 1). For this, multiple views with

FORSCHUNGSFORUM DER ÖSTERREICHISCHEN FACHHOCHSCHULEN 2016
FHK ÖSTERREICHISCHE FACHHOCHSCHUL KONFERENZ
FACHHOCHSCHULE DES BFI WIEN

dynamic query and a Word Tree visual metaphor (Wattenberg & Viegas 2008) similar to ActiviTree (Vrotsou et al. 2009) are used. Thus, the prototype extends a malware analysis environment, providing additional complementary perspectives on the traces. Additionally, the newly created rules can be stored in a knowledge database to support the analysts during their work.

For the design of the prototype, we used a user-centered design process (Sharp et al. 2007). In cooperation with four malware analysis experts, we produced sketches and discussed the appropriate interface design with them. Throughout this step, we created a usage scenario to define the workflow for rule building (Section 2.2). Based on the sketches, we developed a screen prototype (Section 2.1), which was evaluated by an expert review (Section 2.3).

2.1. Interface Design

As IT-security experts are well-skilled programmers, the paper prototype's visual interface design is inspired by the general structure of programming IDEs like Eclipse or NetBeans (Figure 1). As part of the well-known interface structure, we represented the knowledge database (KDB) in the form of a tree structure on the left of the interface, like the project structure in a programming IDE. On the right side of the interface, we positioned the Call Exploration area, like the functions overview area in commonly used programming IDEs. Here, the analyst has the ability to explore all the rules included in the currently loaded analysis file. Additionally, the user gets detailed information about the calls included. Moreover, we integrated the Rule Builder in the center of the screen, usually the development area. This is the most frequently used element of the prototype. Based on this interface structure, we were able to establish a familiar workflow concept on multiple views for the domain experts.

2.2. Usage Scenario

During an analysis task, the IT-security expert found relevant rules in a trace which are currently not included in the knowledge database. He/she switches from the main screen (which is used for the analysis tasks) to the Rule Builder screen, where he/she can build individual rules and integrate them into the KDB (Figure 1). By using the included filters, the expert can reduce the number of calls. (1) He/she selects specific calls in the list and drags them into the main screen. Now he/she gets suggestions for calls, which often occur before or after the newly created rule (Figure 1: Elements above and below the grey construction box). (2) The suggestions can also be used for rule-building by Drag & Drop actions to bring them into the gray box. The number next to the rules built indicates how often the rule occurs in the analysis file. The highlighted calls in the suggestions overview indicate if this specific call is already stored in the rule database. (3) Now, the expert selects a relevant call of the suggestions and reorders the calls by Drag & Drop actions in the box. (4) Then he/she stores the created version of the rule into the KDB. In a final step, the expert switches back to the analysis screen to continue the analysis. The system provides the option of activating or deactivating the rules created by right clicking on a rule or a folder. This way it is also possible to share the expert knowledge generated with other colleagues.

2.3. Expert Review

As domain experts typically do not have expertise in interface design (Lazar et al. 2010), a heuristic evaluation with adapted principles by Nielsen and Mack (1994) and Shneiderman and Plaisant (2009) was part of the evaluation process. To ensure no bias in the results, the evaluation was conducted by two usability experts with 3+ years of experience who were not part of the design team

2.3.1. Design & Procedure

Based on the target group definition and tasks described in the usage scenario (Section 2.2), the usability experts evaluated separately the screen prototype with six adapted design principles suitable for the system:

- The structure and **layout** of the visible information must be comprehensible to the user.
- The workflow, terms and the whole visual design must be **consistent** throughout the entire system.
- **Interactions** can be started, stopped and undone by the user.
- The result of interaction is clearly illustrated by **visual feedback**.
- When using **Drag & Drop**, it is clear how and from where an item can be moved and what the result of the interaction is.
- An **alternative** to Drag & Drop is offered.

Subsequently, the issues found got rated (1 ≔ small issue to 3 ≔ big issue, users will have troubles finishing task) and as a final step, suggestions are provided for the final implementation.

2.3.2. Results

In the following list, we provide a detailed description of the expert review findings (issue with rating number included in parentheses), followed by the suggested solutions for each:

- **Issue**: Unclear how to perform the interaction for ‘undo’ and ‘remove’ (3); **Solution**: Implementation of well-known shortcuts (e.g., to undo Drag & Drop actions → ‘STRG + Z’; to remove calls from the rule → ‘ENTF’).
- **Issue**: The System Calls are represented differently in the interface, thus leading to inconsistency (2); **Solution**: Implementing the same design concept for all kinds of call representations.
- **Issue**: Color coding of already stored calls in the KDB to others is too weak and should not be the only differentiator (2); **Solution**: Providing an appropriate icon and adding a legend in the center display for explanation. Additionally, in the KDB also the icon has to be provided for a clear context.
- **Issue**: Missing visual cues after Drag & Drop interaction (2); **Solution**: While dragging an element to a new spot within the list in the rule builder, the other items should adjust to fit. After storing an element in knowledge database, the current node opens to show the stored rule.
- **Issue**: There is no alternative for Drag & Drop actions for persons with impairments to use the system efficiently (2); **Solution**: Providing a shortcut for the ‘add’ action. Additionally, provide the opportunity to select the elements in the center to get arrows for reordering.

- **Issue**: The meaning of „25“ near the calls in rule builder isn’t clear enough (1); **Solution**: Tooltip and/or label to minimize cognitive workload.
- **Issue**: The histograms in the filtering area have the same coloring as the bars in the call table on the right but they do not represent the same data (1); **Solution**: Usage of different colors for the representation of different data and vice versa.
- **Issue**: It is not clearly visible which area is active in the histogram by the use of the underlying range slider (1); **Solution**: Gray out the bars, which are not active in the histogram and highlight the selected range slider area.
- **Issue**: Calls which are currently included into the newly generated rule should be highlighted (1); **Solution**: Change the font type for such calls to bold.

3. Conclusion & Future Work

Currently, malware analysts are using rules which are constructed by automated analysis methods. In this paper, we presented and evaluated our new rule building screen prototype (CallNet) to support malware analysts during their work on behavior-based malware analysis. One of the benefits of the CallNet prototype is that the analyst gets the ability to construct his/her own rules in a very efficient way due to Drag & Drop actions. These newly created rules will be used for the analysis of potential malicious software samples. All these rules can be included into the KDB. Following the classical interface design paradigm of programming IDEs, the analyst can follow an easy and clear workflow for the design and storage of new rules. Additionally, the suggestions supported by the system represent frequently used calls in relation to the currently constructed rule. To further improve consistency and reduce cognitive workload, an expert review was conducted to identify possible usability issues in an early design stage. By using familiar interaction concepts, the malware analysts can easily experiment with the rules. Furthermore, the expert review suggested implementing expert features like common shortcuts to work more efficiently within the system.

In future work, we plan to implement a functional prototype as a proof of concept including the suggestions of the expert review performed. For the prototype evaluation, we will establish a three-stage user study including rule building tasks (hands-on training for the experts), a system usability scale (SUS) questionnaire and semi-structured interviews.

Acknowledgements

This work was supported by the Austrian Science Fund (FWF) via the KAVA-Time project no. P25489 and the Austrian Ministry for Transport, Innovation and Technology (BMVIT) under the ICT of the future program via the VALiD project no. 845598.

References:

Bazrafshan, Z., Hashemi, H., Fard, S.M.H., Hamzeh, A., 2013. A survey on heuristic malware detection techniques, in: CIKM. pp. 113–120.

Chen, C., 2005. Top 10 unsolved information visualization problems. IEEE CG&A 25, 12–16.

Conti, G., 2007. Security data visualization: graphical techniques for network analysis. No Starch Press, San Francisco.

Dornhackl, H., Kadletz, K., Luh, R., Tavolato, P., 2014. Malicious Behavior Patterns, in: IEEE SOSE. pp. 384–389.

Egele, M., Scholte, T., Kirda, E., Kruegel, C., 2008. A Survey on Automated Dynamic Malware-analysis Techniques and Tools. ACM Comput. Surv. 44, 6:1–6:42.

Kielman, J., Thomas, J., May, R., 2009. Foundations and Frontiers in Visual Analytics. Information Visualization 8, 239–246.

Lazar, D.J., Feng, D.J.H., Hochheiser, D.H., 2010. Research Methods in Human-Computer Interaction. John Wiley & Sons.

Lee, D., Song, I.S., Kim, K.J., Jeong, J., 2011. A Study on Malicious Codes Pattern Analysis Using Visualization, in: ICISA. pp. 1–5.

Nielsen, J., Mack, R., 1994. Usability Inspection Methods, 1 ed. John Wiley & Sons.

Pike, W.A., Stasko, J., Chang, R., O'Connell, T.A., 2009. The Science of Interaction. Information Visualization 8, 263–274.

Sharp, H., Rogers, Y., Preece, J., 2007. Interaction Design: Beyond Human-Computer Interaction, 2. ed. John Wiley & Sons.

Shiravi, H., Shiravi, A., Ghorbani, A.A., 2012. A Survey of Visualization Systems for Network Security. IEEE TVCG 18, 1313–1329.

Shneiderman, B., Plaisant, C., 2009. Designing the User Interface, 5 ed. Addison Wesley, Boston.

Thomas, J.J., Cook, K.A., 2005. Illuminating the Path: The Research and Development Agenda for Visual Analytics. National Visualization and Analytics Ctr.

Trinius, P., Holz, T., Gobel, J., Freiling, F.C., 2009. Visual analysis of malware behavior using treemaps and thread graphs, in: Proc. 6th VizSec. ACM, New York, pp. 33–38.

Vrotsou, K., Johansson J., Cooper., M., 2009. ActiviTree. IEEE TVCG 15(6), 945–52.

Wagner, M., Aigner, W., Rind, A., Dornhackl, H., Kadletz, K., Luh, R., Tavolato, P., 2014. Problem Characterization and Abstraction for Visual Analytics in Behavior-based Malware Pattern Analysis, in: Proc.11th VizSec. ACM, New York, pp. 9–16.

Wagner, M., Fischer, F., Luh, R., Haberson, A., Rind, A., Keim, D.A., Aigner, W., 2015. A Survey of Visualization Systems for Malware Analysis, in: EuroVis (State of The Art Reports). EG, Cagliari.

Wattenberg, M. & Viegas, F.B., 2008. The Word Tree, an Interactive Visual Concordance. IEEE TVCG, 14(6), 1221–1228.

Wegner, P., 1997. Why Interaction is More Powerful Than Algorithms. Comm. ACM 40, 80–91.

Yee, C.L., Chuan, L.L., Ismail, M., Zainal, N., 2012. A static and dynamic visual debugger for malware analysis, in: Asia-Pacific Conference on Communications, pp. 765–769.