Proceedings

Application of Knowledge Methods in Information Security

June 27 – 29, 2022, Smolenice, SK

Editors:
Balogh Štefan
Homola Martin
Mojžiš Ján

https://www.ui.sav.sk/AKMIS/

Editors:

Štefan Balogh

Institute of Computer science and Mathematics, Faculty of electrical engineering and information technology Slovak University of Technology in Bratislava

Martin Homola

Faculty of Mathematics, Physics and Informatics Comenius University in Bratislava

Ján Mojžiš

Institute of Informatics Slovak Academy of Sciences

Supporting project: APVV 1498-2-0220 Ontological representation for security of information systems

ISBN 978-80-974468-0-2

EAN 9788097446802

This research was funded by the Scientific Grant Agency of the Slovak Republic grant number APVV-19-0220.





Table of Contents

This research was funded by the Scientific Grant Agency of the Slovak Republic grant number APVV-19-0220.





PREFACE

Increasing amounts of available data are currently pushing forward almost any area of human interest. In information security such data sources contain knowledge that can be facilitated to improve detection, protection, and timely response against the ever more frequent security threats. The AKMIS Workshop series aim to bring together scientists, researchers, and practitioners to provide an open platform for discussion about recent research developments and future trends in facilitating data and knowledge to improve information security. AKMIS aims at inclusivity, and it only has informal online proceedings.

This volume contains the proceedings from the 2nd AKMIS workshop held in co-location with the 22nd Central European Conference on Cryptography, CECC 2022, in Smolenice, Slovakia, June 27–29, 2022. This year, AKIMS received 10 submissions in the form of extended abstracts. Each submission received 2 reviews from the program committee or form additional reviewers. All 10 submissions turned out relevant to the workshop and so they were accepted and they are included in this volume.

The workshop co-chairs would like to thank the authors, the members of the program committee, and one additional reviewer. We are also grateful to the organizers of CECC 2022 for hosting AKMIS together with their conference at the Smolenice castle. AKMIS 2022 was held under the general patronage of the ORBIS project supported by the SRDA agency under contract No. APVV-19-0220.

Bratislava, December 2022
Štefan Balogh
Ivana Budinská
Martin Homola
Ján Mojžiš

This research was funded by the Scientific Grant Agency of the Slovak Republic grant number APVV-19-0220.

TOPICS

Application of various methods from knowledge extraction, management and artificial intelligence, such as:

- automated reasoning,
- information retrieval,
- data mining,
- machine learning,
- knowledge representation,
- ontologies reasoning, explanation,
- neural networks, argumentation, automated question answering,
- description logics learning algorithms,
- and others

with applications in

- system monitoring,
- information intelligence sharing,
- malware detection,
- malware features analysis,
- ontology based security,
- context reasoning,
- ontology models for security domain,
- incident detections, share schemes and responds strategies,
- and others

The fields of interests of the workshop include topics related to knowledge sharing in various domains. Therefore the topics are not limited to the information security domain.

We would like to thank all our colleagues, authors and participants for their help in organising the workshop, for nice presentations and fruitful discussions and we wish to have chance to organize togehter another workshop AKMIS.

This research was funded by the Scientific Grant Agency of the Slovak Republic grant number APVV-19-0220.

PROGRAM

Monday, 27 Jun

Session 1:

Damas Gruska: Formal methods and system security;

Martin Homola and Júlia Pukancová: Abduction as Diagnostic Tool in Computer Security;

Vasyl Ustimenko and Tymoteusz Chojecki: Graph based secure communications with ontological instruments of knowledge based portal;

Roderik Ploszek: Inductive Logic Programming and Description Logics;

Tuesday, 28 Jun

Session 2:

Peter Švec, Štefan Balogh, Martin Homola and Ján Kľuka: Ontological Representation of the EMBER Dataset;

Štefan Balogh, Peter Švec and Alexander Šimko: Expected results of conceptual learning;

Daniel Trizna and Martin Homola: Towards KB Embedding in Malware Detection;

Štefan Balogh, Peter Švec: Integration of outputs from tools for static and dynamic code analysis into an ontological model;

Round table 1

Session 3:

Ján Mojžiš: The Construction of Rules for EMBER Dataset with the Help of a Decision Tree Model;

Ivana Budinska: Standarts for sharing cyber security indicators and threats;

Round table 2

Wednesday, 29 Jun

Round table 3

This research was funded by the Scientific Grant Agency of the Slovak Republic grant number APVV-19-0220.

Formal methods and system security *

Damas P. Gruska

Institute of Informatics, Comenius University,
Mlynska dolina, 842 48 Bratislava, Slovakia,
gruska@fmph.uniba.sk.

Keywords: security, observations,, information flow, supervisory control, insertion function

Abstract

Formal methods enable us to formulate and hence formally check the systems security from various points of view. A large class of security properties is based on an absence of information flow between public and private or classified states, actions, or any other parameters. It is expected that an attacker has some knowledge about system design and can partially observe its behavior. Basically, we distinguish language-based and state-based security properties, referring to sequences of systems actions or states, respectively. If the system is proved to be insecure; we have several options on what to do next. We can redesign the system. We can express the quantity of information that could leak. We can employ a supervisor controller which prohibits some system's behavior that could reveal classified information, or we can use insertion functions, which by inserting some systems action can prevent leak of classified information. Unfortunately, the most of related properties like security itself, the existence of the supervisor controller, or insertion function are undecidable in general if the underlying computational model has Turing power. Fortunately, for finite-state systems and reasonable limited attackers power, they become decidable. On the other side, in many cases due to state explosion, complexity remains exponential.

* this work was supported by the Slovak Research and Development Agency under the Contract no. APVV-19-0220 (ORBIS).

Abduction as Diagnostic Tool in Computer Security

Martin Homola and Júlia Pukancová

Comenius University in Bratislava, Mlynská dolina, 84248 Bratislava, Slovakia

Abduction [3] is a type of inference, where we have model of some situation or system and we observe some effects that are not supported by this model deductively – abduction looks for hypothetical explanations that can be added to the model in order for the observation to be supported.

To borrow a brief example from a medical domain, assume condition $C1$ causes symptom $S1$ and condition $C2$ causes symptoms $S1$ and $S2$. If we observe $S1$ in a patient, then both $C1$ and $C2$ are plausible explanations for the observation. If, in addition, we observe $S2$, then the explanations narrow down to $C2$.

The model can be of anything, say medical conditions and symptoms [5], manufacturing system [2], or a scene from a sporting event observed by a computer vision software [4].

Abduction offers a flexible framework suitable for diagnosing complex models where there are overlapping symptoms of different conditions, and where one condition may be symptom of another, and so on. Abduction may guarantee to find minimal (or weakest) diagnoses and thus avoid hypothesizing more than it is required [1, 5] which is useful in most application scenarios.

In this presentation we will provide typical examples of abduction applications and we will discuss its possible applications in the area of computer security.

Acknowledgents. This work was partially supported by projects ORBIS, funded by Slovak SRDA agency under contract No APVV-19-0220, KATO, funded by Slovak VEGA agency under contract No 1/0778/18, and by TAILOR, funded by EU Horizon 2020 research and innovation programme under GA No 952215.

References

1. Elsenbroich, C., Kutz, O., Sattler, U.: A case for abductive reasoning over ontologies. In: Proceedings of the OWLED*06 Workshop on OWL: Experiences and Directions, Athens, GA, US. CEUR-WS, vol. 216 (2006)
2. Hubauer, T., Legat, C., Seitz, C.: Empowering adaptive manufacturing with interactive diagnostics: A multi-agent approach. In: Advances on Practical Applications of Agents and Multiagent Systems – 9th International Conference on Practical Applications of Agents and Multiagent Systems, PAAMS 2011, Salamanca, Spain. pp. 47–56 (2011)
3. Peirce, C.S.: Illustrations of the logic of science VI: Deduction, induction, and hypothesis. Popular Science Monthly **13**, 470–482 (1878)

4. Petasis, G., Möller, R., Karkaletsis, V.: BOEMIE: Reasoning-based information extraction. In: Proceedings of the 1st Workshop on Natural Language Processing and Automated Reasoning co-located with 12th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2013), A Corunna, Spain. pp. 60–75 (2013)
5. Pukancová, J., Homola, M.: Abductive reasoning with description logics: Use case in medical diagnosis. In: Proceedings of the 28th International Workshop on Description Logics (DL 2015), Athens, Greece. CEUR-WS, vol. 1350 (2015)

Graph based secure communication with ontological instruments of knowledge base portal.

Vasyl Ustimenko[1,2][0000−0002−2138−2357] and Tymoteusz Chojecki[2][0000−0002−3294−2794]

[1] University of Maria Curie Sklodowska, Lublin 20-036, Poland
[2] University of London (Royal Holloway), UK vasylustimenko@yahoo.pl
[3] University of Maria Curie Sklodowska, Lublin 20-036, Poland
tymoteusz.chojecki@umcs.pl

Abstract. We suggest the scheme of usage of protected knowledge base portal B for secure communica-tion of base administrator (Alice) and public user (Bob).
Assume that the information in B is presented in a binary alphabet. So we can identify characters of this alphabet with elements of finite field F_{256}. Portal has a search engine. So we can assume that the size of the information through the portal is practically unlimited. The ontological instruments like extraction of key words together with graph presented relation of them are available.

Keywords: Ontological extraction, Noncommutative Cryptography, Multivariate Cryptography, families of graphs of large girth, graph based cryptography.

Assume that some secure tools are used to protect the entrance of B. To enter the system user need a password which is a tupple E of length n. We suggest the following access control scheme. Alice and Bob use twisted Diffie-Hellman protocol of Noncommutative Cryptography based on the cubical group $GA(n, F_q), q = 256$ of transformation of vector space $(F_q)^n$ (see author's abstract on CECC 2022 and further references). So, they elaborate multivariate map of kind $x_1 \to f_1(x_1, x_2, \ldots, x_n), x_2 \to f_1(x_1, x_2, \ldots, x_n), \ldots, x_n \to f_n(x_1, x_2, \ldots, x_n)$, where f_i are written via the list of their monomial terms ordered in lexicographical order. The security of this protocol rests on the Conjugacy Power Problem for the group of automorphisms of $F_q[x_1, x_2, \ldots, x_n]$. This problem is on the list of hard problems of Post Quantum Cryptography.

Assume that f_i contains linear form $a(i,1)x_1 + a(i,2)x_2 ... + a(i,n)x_n$. Alice and Bob use vector $E = (f_1(a(1,1), a(1,2), \ldots, a(1,n)), f_2(a(2,1), a(2,2), \ldots, a(2,n)), \ldots, f_n(a(n,1), a(n,2), \ldots, a(n,n))$ to enter the system. Administrator Alice sets E for user Bob and he enters the base B.

Thus, they can use graph based stream cipher C (see [1]) to work with potentially infinite text from $(F_q)^m, m = n^a, a > 1$. The tupple password which encodes two sparse linear transformation and a vector of length m has length $3m - 2$. Alice selects the file D from B and mark the position of the file in

file directory of B. After Bob and Alice use ontological instruments of B to extract file P of 'key words' from D of size $3n^a - 2$ and use stream cipher C for the information exchange. Linearisation attack require the interception of n^{3a} pairs plaintext-ciphertext. So, after the exchange $1/2n^{3a}$ messages Alice has select other piece D' of information from B for creation of new password P' for cipher C. The scheme is implemented in Ukraine for users of "Taras Shevchenko knowledge portal". Recently we modify the stream cipher C to make it resistant to linearisation attacks. The modification is presented in the talk.

References

1. V. Ustimenko, CRYPTIM: Graphs as Tools for Symmetric Encryption, in Lecture Notes in Computer Science, Springer, 2001, v. 2227, 278-287

Inductive Logic Programming and Description Logics*

Roderik Ploszek[0000−0002−3192−0630]

Slovak University of Technology in Bratislava, Slovakia
Institute of Computer Science and Mathematics
roderik.ploszek@stuba.sk

Abstract

Inductive logic programming (ILP) allows new knowledge to be constructed from a given background knowledge with respect to positive and negative examples [2]. ILP uses induction to generate hypotheses that are built on the background knowledge and have to explain the positive examples, but also remain consistent with the negative examples. In other words, based on the background knowledge and observed facts, an algorithm can infer new general knowledge that applies to all observed objects.

Description logics (DLs) are languages that describe some knowledge about a system in the form of a knowledge base. Machine learning algorithms can use this knowledge to reason and induce new knowledge similar to ILP. The main building blocks of DLs are objects, concepts, and roles [1]. In first-order logic, objects are equivalent to constants, concepts to unary predicates, and roles to binary predicates.

The focus of the presentation is to explore the conversion of a program written in PROGOL (which is an ILP system) into OWL (Web Ontology Language) representation that can be processed by DL Learner, a framework for supervised machine learning.

References

1. Lehmann, J.: Hybrid learning of ontology classes. In: International Workshop on Machine Learning and Data Mining in Pattern Recognition. pp. 883–898. Springer (2007)
2. Muggleton, S., De Raedt, L.: Inductive logic programming: Theory and methods. The Journal of Logic Programming **19**, 629–679 (1994)

* This research was sponsored by Slovak Republic under grant APVV-19-0220.

Ontological Representation of the EMBER Dataset

Peter Švec[1], Štefan Balogh[1], Martin Homola[2], and Ján Kľuka[2]

[1] Institute of Computer Science and Mathematics, Faculty of Electrical Engineering and Information Technology Slovak University of Technology, Ilkovičova 3, 81219 Bratislava, Slovakia

[2] Comenius University in Bratislava, Mlynská dolina, 84248 Bratislava, Slovakia

Malware detection is an important problem in information security. Historically, large number of diverse methods have been applied on this problem, including some AI methods such as machine learning [3]. To facilitate research in this area there are several publicly available datasets in this domain such as EMBER [1] and SOREL [2].

In order to apply symbolic AI tools on these data, a suitable ontological representation is required. For instance, Švec et al. [4] were able to to obtain malware characterizations in form of structured concept descriptions, based on an ontology.

We present an updated version of EMBER ontology previously developed by Švec et al. [4]. We expect this updated version to improve the concept learning results. At the same time the ontology was reconstructed using current ontology engineering guidelines and we hope it will be more universal and reusable also by other symbolic or neural-symbolic methods, or any application that needs to process Windows malware related data. The ontology is compatible with both EMBER and SOREL datasets.

Acknowledgents. This work was partially supported by projects ORBIS, funded by Slovak SRDA agency under contract No APVV-19-0220, and by TAILOR, funded by EU Horizon 2020 research and innovation programme under GA No 952215.

References

1. Anderson, H.S., Roth, P.: EMBER: An open dataset for training static PE malware machine learning models. arXiv preprint 1804.04637 (2018). https://doi.org/10.48550/ARXIV.1804.04637
2. Harang, R., Rudd, E.M.: SOREL-20M: A large scale benchmark dataset for malicious PE detection. arXiv preprint 2012.07634 (2020). https://doi.org/10.48550/ARXIV.2012.07634
3. Husák, M., Komárková, J., Bou-Harb, E., Celeda, P.: Survey of attack projection, prediction, and forecasting in cyber security. IEEE Commun. Surv. Tutorials **21**(1), 640–660 (2019)

4. Švec, P., Balogh, S., Homola, M.: Experimental evaluation of description logic concept learning algorithms for static malware detection. In: Proceedings of the 7th International Conference on Information Systems Security and Privacy, ICISSP 2021, February 11-13. pp. 792–799. SCITEPRESS (2021)

Expected results of conceptual learning

Štefan Balogh, [1] Peter Švec [1] and Alexander Šimko [2]

[1] Faculty of Electrical Engineering, Slovak Technical University, Bratislava
[2] Comenius University in Bratislava, Mlynská dolina, 84248 Bratislava

Abstract. In this paper we discuss the result of conceptual learning suitable for use in specific detection systems.

Keywords: conceptual learning, attack detection.

1.1 Future direction

The great efforts of the scientific community are currently focused on increasing the security of information systems, as their dissemination to every area of our lives is an unstoppable process. Attackers and criminals using information weaknesses systems get into different areas of our lives and can use it to their advantage.
Currently, it is necessary to look for solutions that can work automatically as much as possible and are able to respond in a short time to current changes or new attacks.
For fast response and learning, it is necessary to have the information available as soon as possible, which will be included in automated detection systems. This involves depending on the algorithm that provides the automation, either updating the rules or teach the system for new knowledge.
For the necessary acceleration, it would be helpful if a system that shares knowledge could provide data directly to learning systems, or to the systems that generate new rules. It is also possible to consider or test to share not only attack information, but also given detection rules, if they are written in a similar format as shared data about the attack.
More and more work is there, where ontological rules are used for detection [1][2].
If the functionality of such systems will be proven, it would be possible to use conceptual learning to generate the rules.
To be able to use the results of conceptual learning, the resulting rules should be like the rules that the proposed detection systems use. Therefore, it is necessary to analyze in detail the rules of existing detection tools, such as ZEEK OWASP, Snort, Suricata, or mod security [3][4][5][6]. Detection systems that directly use ontological rules in detection can be a valuable resource, also [2][7].

However, great variability and an increase in the number of rules for detection slows down real time detection and the whole process becomes time- or hardware-intensive.. It would be useful here to test the merging of rules into a common decision tree [8][9].
Creating a comprehensive solution for sharing knowledge about attacks and detection rules for these attacks can be a significant step forward in the field of security compared to the current situation.

Acknowledgment : "This research was funded by the Scientific Grant Agency of the Slovak Republic grant number APVV-19-0220."

References

1. Munir, R. F., Ahmed, N., Razzaq, A., Hur, A., & Ahmad, F. (2011, December). Detect HTTP Specification Attacks Using Ontology. In *2011 Frontiers of Information Technology* (pp. 75-78). IEEE.
2. Razzaq, A., Latif, K., Ahmad, H. F., Hur, A., Anwar, Z., & Bloodsworth, P. C. (2014). Semantic security against web application attacks. *Information Sciences*, *254*, 19-38. night, Michelle (2017).
3. Tidjon, L. N., Frappier, M., & Mammar, A. (2020, April). Intrusion detection using astds. In *International Conference on Advanced Information Networking and Applications* (pp. 1397-1411). Springer, Cham.
4. Online https://github.com/schubergphilis/mod_security/tree/master/files/default/owasp-modsecurity-crs-2.2.8
5. Online https://github.com/SpiderLabs/owasp-modsecurity-crs/tree/v3.3/dev/rules
6. Online https://github.com/coreruleset/coreruleset/tree/v4.0/dev/rules
7. Ulicny, B. E., Moskal, J. J., Kokar, M. M., Abe, K., & Smith, J. K. (2014). Inference and ontologies. In *Cyber Defense and Situational Awareness* (pp. 167-199). Springer, Cham.
8. Abdelhalim, A., Traore, I., & Nakkabi, Y. (2016). Creating Decision Trees from Rules using RBDT-1. *Computational Intelligence*, *32*(2), 216-239.
9. Khan, Z. M. A., Saeidlou, S., & Saadat, M. (2019). Ontology-based decision tree model for prediction in a manufacturing network. *Production & Manufacturing Research*, *7*(1), 335-349.

Towards KB Embedding in Malware Detection

Daniel Trizna and Martin Homola

Comenius University in Bratislava, Mlynská dolina, 84248 Bratislava, Slovakia

The problem of *malware detection* is to classify logged data about processes and their behaviour and to identify which processes are potentially dangerous (malware). Historically, large number of diverse methods have been applied on this problem, including some AI methods such as machine learning [3].

Knowledge base (KB) embedding is a neural-symbolic approach to represent knowledge base in a low-dimensional vector space keeping as much essential information as possible. In recent years there have been many approaches to achieve such representation ranging from simpler attempts like TransE [2] to more sophisticated ones like Cone embedding [5] and Sphere embedding [4].

We propose to apply KB embedding to the problem of malware detection. We will work with the EMBER dataset [1] and its ontological representation developed by Švec et al. [6, 7]. The main advantages we hope this approach could give us are the following:

- the approach is combinable with standard machine learning algorithms once we have good embedding;
- the embedding could reveal hidden connections in our dataset;
- the embedding may possibly improve detection results due to incorporated symbolic knowledge.

Acknowledgents. This work was partially supported by projects ORBIS, funded by Slovak SRDA agency under contract No APVV-19-0220, and by TAILOR, funded by EU Horizon 2020 research and innovation programme under GA No 952215.

References

1. Anderson, H.S., Roth, P.: EMBER: An open dataset for training static PE malware machine learning models. CoRR abs/1804.04637, arXiv.org, https://arxiv.org/abs/1804.04637 (2018)
2. Bordes, A., Usunier, N., Garcia-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data p. 2787–2795 (2013)
3. Husák, M., Komárková, J., Bou-Harb, E., Celeda, P.: Survey of attack projection, prediction, and forecasting in cyber security. IEEE Commun. Surv. Tutorials **21**(1), 640–660 (2019)
4. Kulmanov, M., Liu-Wei, W., Yan, Y., Hoehndorf, R.: EL embeddings: Geometric construction of models for the description logic EL ++. CoRR abs/1902.10499, arXiv.org, https://arxiv.org/abs/1902.10499 (2019)
5. Özçep, Ö.L., Leemhuis, M., Wolter, D.: Cone semantics for logics with negation. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020. pp. 1820–1826 (2020)

6. Švec, P., Balogh, S., Homola, M.: Experimental evaluation of description logic concept learning algorithms for static malware detection. In: Proceedings of the 7th International Conference on Information Systems Security and Privacy, ICISSP 2021, February 11-13. pp. 792–799. SCITEPRESS (2021)
7. Švec, P., Balogh, S., Homola, M., Kľuka, J.: Ontological representation of the EMBER dataset. In: Procs. of AKMIS 2022, The 2nd workshop on Application of Knowledge Methods in Information Security (to appear) (2022)

Integration of outputs from tools for static and dynamic code analysis into an ontological model

Štefan Balogh, [1] and Peter Švec [1]

[1] Faculty of Electrical Engineering, Slovak Technical University, Bratislava

Abstract. In this paper we discuss the integration the outputs of analysis tools into an ontological model.

Keywords: dynamic analysis, ontological model.

1 Tools for static and dynamic code analysis

1.1 Holmes processing

The designed system for static and dynamic analysis allows you to enter samples via a web interface or script, automatically or manually run static and dynamic analysis, save samples, display results, and add results to the ontology. The Holmes Processing project was used as the basis of the system [1]. Five modules were selected from the Holmes Processing project, namely Gateway, Storage, Totem, Totem-Dynamic, Interrogation and Frontend. They have been combined into one project to facilitate installation.

1.2 Static analysis

The system uses services running using the Docker tool for static analysis. For the demonstration of the solution was created one static service that obtains information about Portable Executable (PE) files. For analysis, the Python module called pefile is used, with which it is possible to extract data from the header from executable files such as: imported libraries, strings, sections and other data. It also allows you to determine whether packer or obfuscation was used when creating the file based on the entropy of each data section. The result of the analysis is transformed into Mid-Atlantic Equity Consortium (MAEC) format so that it can be added to the ontology and sent directly to the REST service for conversion to Web Ontology Language (OWL). A pefile-to-maec library [2] from Mitre was used for data extraction and conversion. It was necessary to transform the output to MAEC 5.0 because the library only supports conversion to MAEC 4 in XML format. The service uses the pefile Python library to obtain all values from the PE header and maps the key names to MAEC format. Then the resulting python dictionary is converted to JSON in MAEC format according to the Structured Threat Information eXpression (STIX) definition. Data such as size, data sections, their entropy and optional headers such as linker version, operating system version and the like are selected from the pefile to the MAEC.

1.3 Dynamic analysis

The system uses two tools for dynamic analysis. Cuckoo [3], which is one of the most popular tools, and Drakvuf [4], which is difficult for malware to detect. The existing Cuckoo Sandbox service in the Holmes system did not support obtaining results in MAEC format [5]. It was therefore necessary to add this functionality to the service. The Cuckoo tool itself also had to be modified. Cuckoo only supports MAEC results with a Mitre plug-in.

The Drakvuf Sandbox is used to work with the Drakvuf analyzer. For connecting the Holmes system to the Drakvuf Sandbox a separate service was created. Its function is similar to the service for the Cuckoo tool, which means that it is used as a Docker container, which offers a REST API for communication with Holmes and queries the Drakvuf Sandbox tool where it creates new tasks, finds out the status and obtains results. In addition, it also processes the obtained results and creates a valid JSON document from them in a format similar to Cuckoo. Thanks to the use of the same output format as Cuckoo, it is possible to use a modified tool from the Mitre company to convert it to MAEC format. which gives almost the same results as the conversion from Cuckoo. The same results in the conversion are necessary to combine the outputs when saving to the ontology. To use the same conversion tool, the service had to be implemented in Python, unlike the Cuckoo service, which is in Go. The service creates HTTP queries on the Drakvuf Sandbox API which allows it to create a new task, find out the status of the task and get the results of the analysis.

2 Ontology model

To store MAEC data obtained during analysis in ontology, it is first necessary to convert them to OWL format. The custom Java application was used to do it. Supported objects from the MAEC standard are malware-instances and malware-actions and they share one MAEC Object class. One ObservableObject data class was created from the STIX standard for all objects, but not all properties but only those used in the ontology. After saving STIX objects, MAEC objects are converted. The MAEC objects are stored in a worksheet through which it is iterated, and the conversion method is called according to the object type. Currently, only malware-action and malware-instance types are supported. The malware instance is identified by the md5 hash of the STIX object that is its instance. Although there may be multiple instances, each must have in MAEC specification the same hash. The individual is therefore created from the hash of the first instance. In a malware action, the individual identifier in the ontology is created first. The identifier consists of hashes of input STIX objects (input_object_refs), hashes of output STIX objects (output_object_refs) and the action name. These values are combined into a single string to form an md5 hash. In a malware action, the name of the action, input STIX objects in the input relation, and output STIX objects in the output relation are stored in the ontology. The counter is in a "do" relationship with the process, such as “process to counter” and in relation "count" with class

Malware_Action, such as “counter count action.”. The counter identifier is created from the md5 hashes of the malware action and instance. These are connected into one chain and are of them created an md5 hash and added the word Counter at the end, for example “a00e5 ... Counter”. A role “initiated_malware_action” has also been added to the ontology to directly link process and action. The identification of individual elements of the ontology is as follows:

• Malware_Instance - md5 hash of the STIX object.
• Malware_Action - md5 hash from md5 hashes of all input and output objects combined into one string and action name.
• ObservableObject - Either an existing md5 hash file or an md5 hash from a string created by combining a type, name, path, command line, key, and mime type.
• Counter - an md5 hash created from a string joined from an md5 hash action and an md5 hash instance.

After saving instances and actions, the processes obtained from the dynamic analysis are saved. It goes through the list of all stored instances and its processes are connected to each via OWLObjectProperty instance-process. It is handled that the method for saving processes is skipped if the sheet is empty, so if it has no processes. Each process connects only once, even if it is called multiple times, because the identifier consists of a hash of the detected data.

The conversion speed was tested on two files, the first with 28729 malware actions and 109 STIX objects and the second with 15668 actions and 1117 objects. Conversion of the first file in about 2 seconds. The conversion time of the second file was approximately 5 seconds. Application optimization brought the required conversion acceleration.

Acknowledgment : "This research was funded by the Scientific Grant Agency of the Slovak Republic grant number APVV-19-0220."

References

1. WEBSTER, G., HANIF, Z., LUDWIG, A., LENGYEL, T., ZARRAS, A., and ECKERT, C. Skald: A scalable architecture for feature extraction, multi-user analysis, and real-time information sharing. vol. 9866, pp. 231–.
2. The MITRE Corporation. pefile-to-maec . online: https://github.com/MAECProject/pefile-to-maec, 2017.
3. Oktavianto, D., & Muhardianto, I. (2013). *Cuckoo malware analysis*. Packt Publishing Ltd.
4. LENGYEL, T., MARESCA, S., PAYNE, B., WEBSTER, G., VOGL, S., and KIAYIAS, A. Scalability, fidelity and stealth in the drakvuf dynamic malware analysis system.
5. The MITRE Corporation. Maec - malware attribute enumeration and charac-terization [online]. [cit. 29.03.2020]. online: https://maecproject.github.io/, 2020.

EMBER Dataset Analysis from The Machine Learning Point of View

Ján Mojžiš[1][0000-0002-2196-2271]

[1] Institute of Informatics, SAS, Bratislava, Slovakia
jan.mojzis@savba.sk

Abstract. Ontology is traditionally used in concept learning. Decision tree models were used before in the process of ontology creation, in applications of recommender systems or for prediction in a manufacturing network. Decision tree model in comparison to neural networks can generate interpretable rules based on the features located in dataset. In this paper, a decision tree model, which was created with the Waikato Environment for Knowledge Analysis (WEKA) application, is used to generate rules from EMBER dataset. In the result, three rules are generated, covering 52.6% of all malware samples presented in the dataset.

Keywords: Ontology, decision tree, EMBER, malware.

1 Introduction

An ontology is a formalized, structured and machine-readable representation of data. A learning software can be used to process it and to discover the rules or concepts contained in the data. In concept learning, we use the software learner to discover concepts or rules that should describe the data and relations between entries. A traditional machine-learning practices can be used to evaluate the features in the ontology data. Decision trees can help with rules or concepts constructions, finding appropriate relations and features. In this paper we apply J48 decision tree approach to ontological data, discovering three rules, covering more than 50% of all malware entries, contained in the dataset.

2 Related works

Zalan et al. [3] propose a predictive model, which assist in the allocation of newly received orders in a manufacturing network. The methodology presents the mapping of a PROSA (Product-Resource-Orcer-Staff Architecture) based ontology on a decision tree, created with the Waikato Environment for Knowledge Analysis (WEKA) application [1]. A decision tree algorithm was mapped into ontology with the help of Semantic Web Rule Language (SWRL). The SWRL rules were extracted from the

decision tree model with the help of a MATLAB program. In the result a model gave 60.4% prediction accuracy.

Bouza et al. [2] propose SemTree for the use in the recommender system. SemTree is an ontology-based decision tree learner, that use a reasoner and an ontology to semantically generalize item features to improve the effectiveness of the decision tree built. He evaluates SemTree with J48 and outperforms it [1].

In the objective of this paper the ontology is already given, and I search for the rules which should help to discriminate malware entries from benign entries.

3 Dataset and ML model used

I have used Weka's J48 Tree model in Weka [1]. The dataset EMBER [4] is a collection of statically extracted features from approximately 1.1 million benign/malicious Windows executables. I have reduced the original dataset to 500 samples of malware and 500 samples of benign-type records. Features were filtered with the help of CFS [5] feature selector, available in Weka.

4 Results

The tree model generated the following rules:

1. #pe_high_entropy > 1, data <= 7.246707 (137 of which 6 is false)
2. #pe_high_entropy <= 1, code > 6.522143, #pe_registry > 0 (54 of which 2 are false)
3. #pe_high_entropy <= 1, code <= 6.522143, #pe_dll <= 0, data > 3.799375, data <= 4.083352, rsrc <= 5.614608, rdata > 5.0436 (72 of which 1 is false)

The rules 1. and 2. can be seen in visualized tree in Fig. 1.

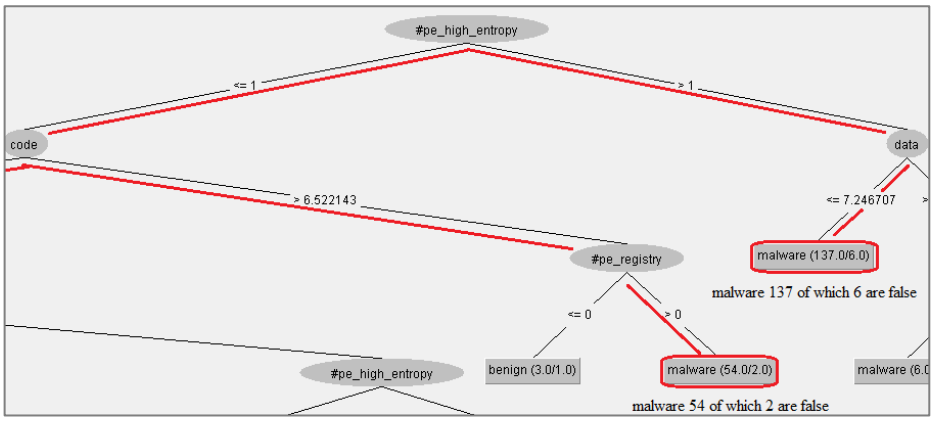

Fig. 1. An excerption of the generated J48 Tree from the sample dataset.

5 Conclusions

In this short paper the application of decision tree on EMBER dataset is presented. Three rules were generated which together are covering 263 out of 500 malware samples, contained in the reduced dataset. Reduced dataset proved that there are rules discriminating malware and benign samples.

Acknowledgment

This work was supported by the following research grants VEGA 2/0155/19 and APVV-19-0220.

References

1. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl.: Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web, ACM, Hong Kong (2001).
2. Bouza, A., Reif, G., Bernstein, A., Gall, H.: SemTree: ontology-based decision tree algorithm for recommender systems. In: International Semantic Web Conference, Karlsruhe, Germany (2008).
3. Khan, Z.M.A., Saeidlou, S. and Saadat, M.: Ontology-based decision tree model for prediction in a manufacturing network. Production & Manufacturing Research *7*(1), 335-349 (2019).
4. Anderson, H. S. and Roth, P.: Ember: an open dataset for training static pe malware machine learning models. arXiv preprint arXiv:1804.04637.
5. Hall, M.A., 1998.: Correlation-based feature subset selection for machine learning. Thesis submitted in partial fulfillment of the requirements of the degree of Doctor of Philosophy at the University of Waikato (1998).

Standarts for sharing cyber security incidents and threats

Ivana Budinská[1]

[1] Institute of Informatics, Slovak Academy of Sciences, Dubravska cesta 9, 845 07 Bratislava, Slovakia

budinska@savba.sk

Internet is recognized as a global critical infrastructure. Since first internet worms have appeared, the need for sharing information about incidents and threats has been constantly increasing. Sharing information about cyber security incidents helps professionals and users of the internet to cope with attacks, and to prepare and protect their systems against known attacks. FIRST, the Forum of Incidents Response and Security Teams has been established in 1990 with the aim to bring together experts from cyber security response teams to cooperate on handling security incidents, to share information about vulnerabilities and threats. FIRST is also very active in publishing standards related to the cyber security. Among the most commonly used systems operated by FIRST belongs CVSS: Commmon Vulnerability Scoring System that provides standards on the characteristics and severity of softeare vulnerabilities. Another broadly used system MISP (malwareinformation sharing platform). Besides of a sw platform for collecting, storing, distributing and sharing malware information, it provides also a great number of standards that help sharing info with humans and automated systems. The MISP core format is based on JSON. The other standards address various template that are used to construct MISP objects, taxonomy format for description of machine (triple) tag vocabularies, galaxy format for attaching additional information such as MISP events or attributes, and a SightingDB format that helps to give automated context to attributes (e.g. by counting occurrences and tracking times of observability). A new standard is being prepared – MISP warning list format.

The presentation will provide a brief introduction to MISP standards.

This work was supported by project No APVV-19-0220

References

1. Regulation (EU) 2019/881 of the European Parliament and of the Council of 17 April 2019 on ENISA (the European Union Agency for Cybersecurity) and on information and communications technology cybersecurity certification and repealing Regulation (EU) No 526/2013 (Cybersecurity Act) https://eur-lex.europa.eu/eli/reg/2019/881/oj.
2. MISP – A threat sharing platform, User Guide, https://www.circl.lu/doc/misp/book.pdf