Capturing Malware Behaviour with Ontology-based Knowledge Graphs

Ipshita Roy Chowdhury and Deepayan Bhowmik
Division of Computing Science & Mathematics, University of Stirling, Stirling, FK9 4LA, UK
{ipshita.roy.chowdhury, deepayan.bhowmik}@stir.ac.uk

***Abstract*—Exponential rise of Internet increases the risk of cyber attack related incidents which are generally caused by wide spread frequency of new malware generation. Different types of malware families have complex, dynamic behaviours and characteristics which can cause a novel and targeted attack in a cyber-system. Existence of large volume of malware types with frequent new additions hinders cyber resilience effort. To address the gap, we propose a new ontology driven framework that captures recent malware behaviours. According to code structure malware can be divided into three categories: basic, polymorphic and metamorphic. Packing or code obfuscation is also a technique adopted by the malware developers to make the code unreadable and avoid detection. Given that ontology techniques are useful to express the domain knowledge meaningfully, this paper aims to develop an ontology for dynamic analysis of malware behaviour and to capture metamorphic and polymorphic malware behaviour. This will be helpful to understand malicious behaviour exhibited by new generation malware samples and changes in their code structure. The proposed framework includes 14 malware families with their sub-families and 3 types of malware code-structure with their individuals. With a focus on malware behaviour the proposed ontology depicts the relations among malware families and malware code-structures with their respective behaviour.**

***Index Terms*—Ontology, Malware, Metamorphic, Polymorphic, Packing**

I. Introduction

Due to the high reliance on computer networks and information technologies many organizations and companies have become targets for cybercriminals. Because of ubiquitous nature of Internet and cyber systems there are always new emerging threats that are causing significant g rowth o f new generation malware and attack variants. Web applications, mobile platforms and social networking cites are constantly making the end-users highly vulnerable to malware attacks.

There exists a wide variety of malware types, including Trojan horses, ransomware, viruses, spyware, adware, worms, DDoS, zombies, backdoors, and so on. According to all reports, cyber-attacks are becoming more sophisticated Undoubtedly Trojans, in terms of functional variety and ability to propagate them, are all-purpose weapon for malware developers than any other classes of malware. In 2018, cyber criminals developed 62.51% Trojans for Windows, 21.06% classic computer viruses and 6.62% Internet worms. According to McAfee Labs report the number of malicious files increased to 79 million per day in 2018. In the same year a malware named Mirai led the ranking by accounting 41.19% of the overall malicious code for IoT devices whereas AV test report indicates a constant distribution of Windows malware and PUA (Potentially Unwanted Applications).

In recent times because of the exponential development of Internet-based systems [1] in users' daily life damage caused by potentially harmful new malware generation becomes more frequent. It's crucial to protect the end-users and software developers from malware infection. Because malware writers always adopt new advanced techniques like, polymorphism, obfuscation and packer to avoid detection. Polymorphic malwares are designed to change it's own code using the polymorphic engine but retains a part of it's original code that remains same in the following version whereas metamorphic malwares completely rewrite itself so the new version no longer matches the previous iteration (makes it difficult to recognize than polymorphic malware) [2].

Ontologies have been used to conceptualize the domain knowledge of malicious behaviour properly. Different Ontology-based frameworks have been developed for malware detection and prevention. Modeling malware behaviour is an important task to understand complex malware behaviour. [3] and [4] are two works related to modeling malware behaviour using ontologies. Suspicious samples and benign samples are distinguished based on their exhibited behaviour. In [5] Kiwia *et al.* (2018) proposed a cyber-kill chain based taxonomy of banking trojans to improve the mitigation techniques.

Literature suggests existing ontology-based frameworks that helps to detect malware [6]–[9]. They cover a range of activities including analyzing malware behaviour, mobile malware (*e.g.*, Android) or malware knowledge base. However, detection of new generation malware is a challenging task because of advanced anti-detection techniques and ability to change the malicious code in every iteration (polymorphism and metamorphism). Behaviours of polymorphic and metamorphic malware are difficult to detect because of their complex and dynamic nature. Ontology-based frameworks should be dynamic to adapt changes, to add new instances and to include new datasets which expresses new generation malware behaviour.

Common malware detection methods such as signature-based detection techniques mostly rely on human expertise to create the signatures in detecting a malicious behaviour in the code. The detectors look for the previously defined signature in the code. However the major drawback of the

signature-based method is that it cannot detect new type of attacks like, zero-day [10]. Anomaly-based detection technique uses its knowledge to distinguish the deviation between normal and malicious behaviour during program inspection (used in Intrusion Detection Systems(IDSs)). Such techniques characterize the normal behaviour and identify attacks based on deviations from normalcy [11]. The major shortcomings of anomaly-based malware detection are high false alarm or false positive rate, time complexity and difficulty in feature selection for training phase [10]. Insufficient expertise and knowledge about ever-changing malware behaviour also cause new malware attack.

To address these gaps this paper aims to incorporate ontology to explore domain knowledge of malware, create a dynamic ontology-driven framework to include new generation malware features from a current dataset. To build the proposed ontology this paper focuses on a large literature survey [12], [13], [14], [15] and reports[1],[2] to gather information about new generation malware samples and their exhibited behaviour. Our contributions on this paper are:

- This paper proposes a base ontology structure which divides Malware in two sub-categories: Malware Families and Malware Code-Structure. Malware Families have 14 sub-classes. The sub-classes are included with instances or individuals.
- A benchmark dataset [16] for metamorphic malware was used as the base ontology which includes the behaviour of metamorphic malware depending upon the changes in API call sequences. In the proposed framework the output ontology captures the sequences of API calls (behaviour) where the sequences of API calls were added as 'Individuals' in this base ontology. The output ontology can be viewed and comprehend through a HTML website.
- As part of the evaluation, reasoning of the the output ontology was done through 'HermiT' reasoner. The output file can be exported in .owl or .rdf format and can also be viewed through a HTML.

II. Related Work

Within the scope of this paper, we have split our related work in two subsections, 1) modelling of malware behaviour and 2) ontology-based frameworks.

A. Modelling Malware Behaviour:

Proper understanding and comprehension of domain knowledge of malicious behaviour are required to mitigate this problem. A Malware Behaviour Ontology (MBO) was proposed in [3]. Grégio *et al.* (2016) proposed an ontology to model the knowledge of malware behaviour by using over two thousand malicious samples and almost 400 benign samples to test the rules of suspicious behaviours. Another work was found in [4] where Grégio *et al.* (2014) proposed a malware ontology based on their exhibited behaviour to identify the unknown malware samples and to distinguish the suspicious program from the benign. This work was also done to model authors knowledge of malware behaviour to find a solution from new generation malware samples which can easily compromise the detection techniques of users systems.

Malware detection is a challenging task in financial industries. Detection of banking Trojans is a great challenge due to advanced anti-detection techniques like obfuscation. A cyber-kill chain based taxonomy of banking Trojan features proposed in [5] to improve the mitigation techniques. Kiwia *et al.* (2018) proposed the CKC-taxonomy of banking Trojan features based on the evolutionary computational intelligence. In this work 127 banking Trojans from December 2014 to January 2016 of a UK-based financial organisation have been used to validate the taxonomy. In [17] a malware ontology was proposed to represent analyzed malware characteristics and their relationships. A semantic relation mapping was presented and used in semantic search engines. A malware fuzzy ontology was developed to describe the malware relationships. MALOnt [18] is an open-source malware threat intelligence ontology for knowledge-graph generation and information extraction. An OWL-based malware analysis ontology was built in [19]. A malware analysis dictionary and taxonomy were built and by combining these two a competency model was developed to create an ontology-based competency framework.

Lack of well structured database is a challenge in cyber security field and domain knowledge needs to be semantically structured in Information Security field also. Iannacone *et al.* (2015) proposed an ontology in [20] to combine various publicly-available datasets with internal information for the analysts and automated tools in order to overcome the lack of structured datasets. Ontology of network and computer attacks are also essential to understand the attack pattern. In [21] an ontology of denial of service attack was developed using Protégé software. The accuracy ofthe ontology was tested using Racer software and KDD cup99 test dataset.

B. Ontology-based Frameworks:

Huang *et al.* (2010) proposed an ontology based intelligent system named Taiwan Malware Analysis Net (TWMAN) in [6] to analyze malware behaviour. The behaviour information was stored in ontology repository which is used by the malware behaviour analysis agent and ontology agent to keep the system safe from Viruses and Trojans. Detection of malware by its behaviour was demonstrated in [9] using Ontologies and rules. Infected computer systems were used to develop a host level detection mechanism to identify obfuscated malware codes. Different automated malware detection and prevention approaches were developed previously.

Mobile devices are the repository of users' financial information, social networking activities, banking and emailing. Mapping the relationship among permissions, malware, and benign apps is troublesome and cannot be done manually. To facilitate the application testing in Android ecosystem an ontology-based framework was developed in [7] by Navarro *et al.* (2018) to map the relationship between application and

[1]https://searchsecurity.techtarget.com/definition/metamorphic-and-polymorphic-malware

[2]https://www.lastline.com/blog/polymorphic-malware-real-life-transformers/

system and a machine learning framework to analyze the malware features. Chiang *et al.* (2010) proposed an ontology based approach for mobile malware behaviour analysis in [8] to provide information about a new mobile malware signature in a proper time to the users and organizations to keep their mobile devices safe when the device is not able to obtain the information because limited resources and outdated malware signature within the device.

An ontology-based intelligent model for mobile malware detection was build in [22]. Based on the static features an Apps Feature Ontology (AFO) was built. A concept vector was created and features were selected using optimization algorithms. In [23] ontology was used to develop a model which can detect attack profile of a malware also result of a targeted attack to be successful in mobile security. An advanced semantic decision making system was proposed in [24] to identify malicious programs. In this work integration of semantic technologies and computational intelligence methods was done by integrating the Fuzzy Ontologies and Fuzzy Markup Language (FML). Obrst *et al.* [25] developed a cyber ontology architecture based on an initial malware ontology to construct a semantic model in cyber security domain. This architecture composed of three levels domain ontologies, middle-level and upper level ontologies.

An automated system of malware analysis and evasion detection was proposed in [26]. In this study AEMS (Analysis Evasion Malware Sandbox) was developed using ontologies and MAEC (Malware Attribute Enumeration and Characterization). This architecture consists of detection of malware evasion technique and a countermeasure to force the malware for expressing its complete behaviour. In [27] a dynamic malware behaviour detection was established based on system calls by studying APT malware. Here an ontology-based knowledge framework was built to represent malicious behaviour.

III. Methodology[3]

To address the gap in the current literature this research aims to contribute a malware ontology to capture malware behaviour. As a starting point, we've considered a malware dataset that is based on API call sequence of Windows PE (Portable Executable) [16] and used it as the basis to create and visualize a new ontological structure with it's reasoning. Our aim is to provide ontology output to include new individuals from the dataset as ontology individuals are much easier to handle than classes [28]. In this case, object properties were defined to establish relation between the classes and data properties were defined to connect the classes to the literals. While details of the the ontology is given below, following steps described the development process:

1) The Ontology has three main classes: Malware Family, Malware Code Structure and Behaviour.
2) Malware Code Structure has five sub-classes: Basic, Polymorphic, Metamorphic, Packing and Code-obfuscation. Where Packing and Code-obfuscation are equivalent to each other.
3) The class 'Malware Family' has 14 sub-classes.
4) The Behaviour class has one sub-class i.e. "Changes in Code". This sub-class has two child classes: 'Encryption and Decryption', 'Translation and Rewriting'.
5) Relation between 'Behaviour' and 'Malware Code Structure' was declared with an object property named 'shows-Behaviour'.
6) Relation between classes, sub-classes with literals was defined with data properties.

[3]Proposed Ontology can be accessed through https://github.com/ipsrychow/Malware-Ontology-Graph

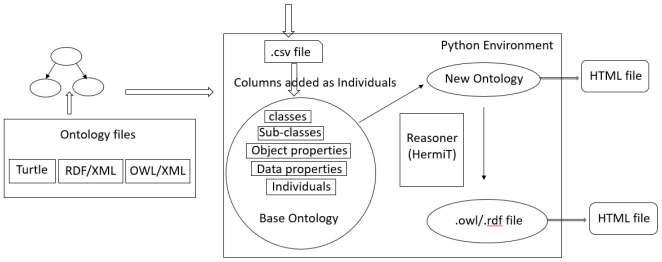

Fig. 1: Overall Workflow

The ontology has been developed using Protégé OWL editor [4]. Protégé is a graphical user interface to build ontologies. The objective is to use this ontology to create a dynamic web-based ontology which will be able to add new characteristics from a given dataset. The overall framework is depicted in Fig. 1.

A. Ontology Engineering

As part of the ontology engineering the domain ontology was created by defining classes, sub-classes, data properties, object-properties, instances etc. The domain ontology consists of a superclass: Thing. The superclass has four sub-classes. The classes are connected with relations or object properties among them. In this ontology, classes were defined based on malware behaviour, types of malicious software, malware families and malware code structure. The overall structure is shown in Fig. 2.

There are 14 types of malware families were included to reflect recently available malware generations. Malware Code Structure includes five classes: *Basic*, *Metamorphic*, *Polymorphic*, *Packing* and *Code Obfuscation*. The ontology shows Packing and Code obfuscation are equivalent to each other. The Behaviour class of the ontology is showing a specific behaviour *i.e.*, Changes in Code. In the behaviour class, encryption and decryption is a behaviour of polymorphic malware and translation and rewriting is a behaviour of metamorphic malware: this relation has been defined using an object property "showsBehaviour". Within the sub-property 'MetamorphicBehaviour' is for the Metamorphic malware and 'PolymorphicBehaviour' is for the Polymorphic malware. Finally as the validator we used a reasoner (HermiT) to produce a log file in Protégé indicating the framework's consistency.

[4]https://protege.stanford.edu/

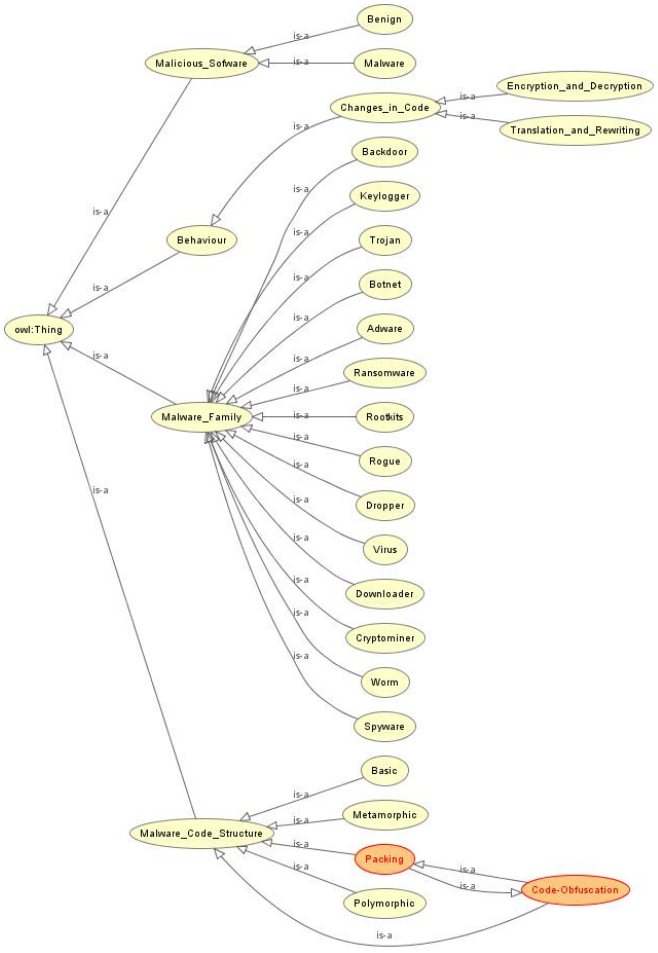

Fig. 2: The base (or domain) Ontology

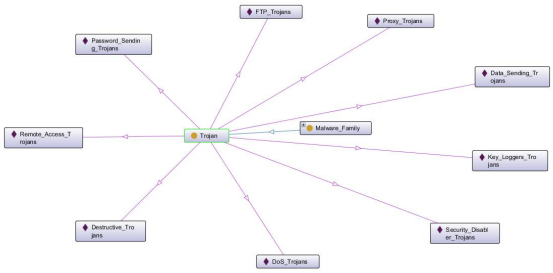

Fig. 3: Individuals of Trojan

Within the sub-classes of the class 'Malware Family' are joined to the respective individuals. For example, Trojan has nine individuals [29] as shown in Fig. 3. Instances of other sub-class individuals of 'Malware Family' are shown in Fig. 4. Currently the ontology contains five object properties and five properties. Object properties include *hasClassification*, *hasCodeStructure*, *malclassification*, *second* and *showsBehaviour* to establish connection between two classes. The data properties establish the relations between classes and literals and they are *classification*, *requiredTime*, *hashValue*, *millisecond* and *hasOpCode*.

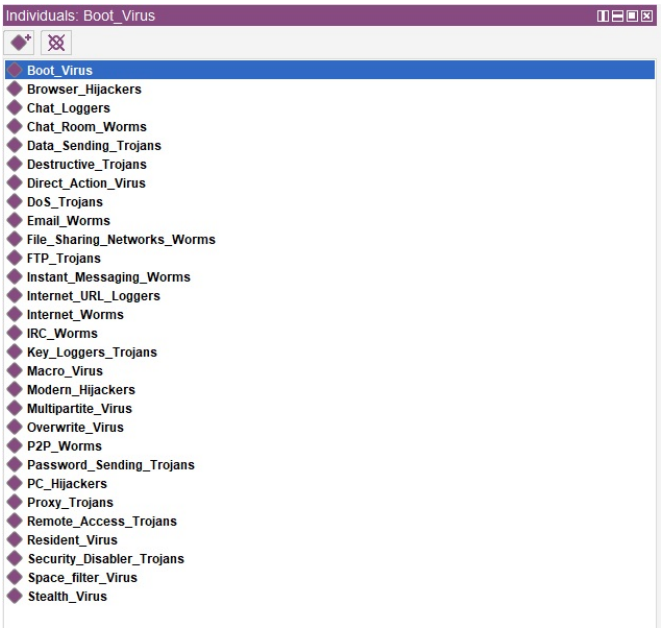

Fig. 4: Individuals of other classes

Visualization of the whole ontology including classes, subclasses, data properties, object properties and instances can be viewed in Fig. 5. This structure was build using Graphical Ontology Editor: OWLGrEd [5].

B. Ontology-oriented programming

To establish an object-oriented approach ontology-oriented programming was designed to include new dataset in this domain ontology which has been used as a base ontology. A novel malware behaviour dataset has been collected from GitHub and included into this ontology to produce a new ontology. Ontology-oriented programming connects the classes, properties and instances of an ontology to a programming language [28]. In *static* approach of ontology oriented programming source code is generated from an ontology itself. Because of static nature of this code it cannot be object-oriented and automatic. On contrary the *dynamic* feature of an ontology facilitates data exchange, changes in data structure and automatic changes in real-time. It provides a 'reasoning engine' to check the consistency of the ontology. Dynamic models are much easier to accept changes in an ontology from it's background. In this study Python was used to carry out the ontology-oriented programming, because it's dynamic and object-oriented nature. In this approach, classes were defined from each class of the ontology. The process is same to define the object-properties and data-properties also. The classes of the ontology are shown in Fig. 6.

[5]http://owlgred.lumii.lv/

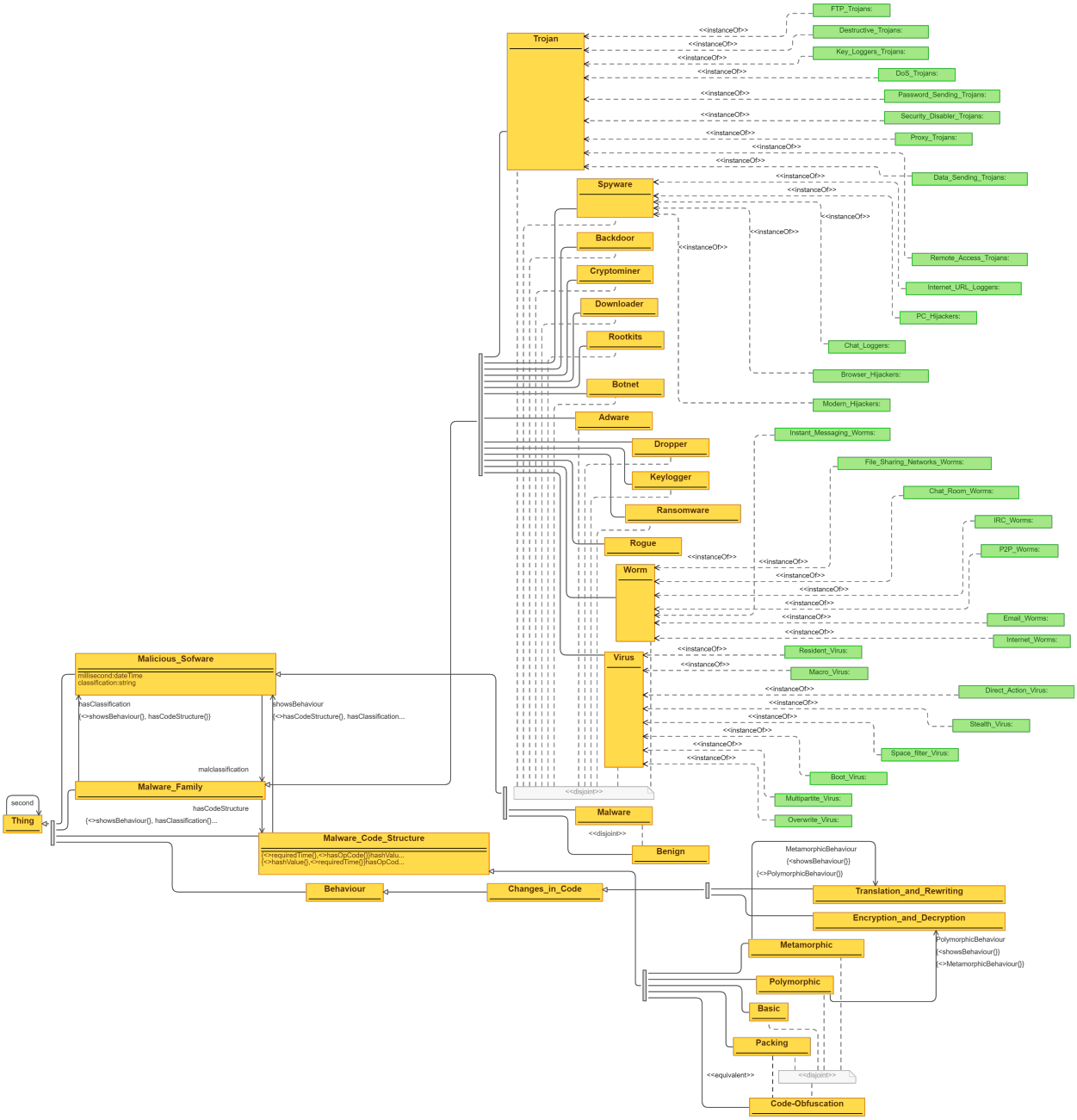

Fig. 5: Overall visualization of the ontology

A dataset [6] was collected from GitHub to import into this base ontology. The sequences of API calls of the dataset were selected to add into the ontology as 'Individuals' with a data property 'hasOpCode'. The output ontology was generated using the dataset including the individuals. Exmaples of such individuals are shown in Fig. 7.

[6]https://raw.githubusercontent.com/ocatak/malware_api_class/master/sample_analysis_data.csv

IV. Evaluation of the Ontology

In order to evaluate our ontology framework we have undergone a series of steps to measure the quality and competency of the ontology. Such evaluation also helps to compare the proposed ontology with other ontologies of this field. For the benefit of the reader we start with describing some of some terminologies of ontology metrics:

1) **Class Complexity:** Average number of paths to reach a class from the superclass *Thing*.
2) **Property Complexity:** Average number of semantic re-

```
#Ontology Classes

behaviour_class = URIRef(base_iri + 'Behaviour')

change_class = URIRef(base_iri + 'Changes_in_Code')

encrypt_class = URIRef(base_iri + 'Encryption_and_Decryption')
translate_class = URIRef(base_iri + 'Translation_and_Rewriting')

software_class = URIRef(base_iri + 'Malicious_Software')

malware_class = URIRef(base_iri + 'Malware')
benign_class = URIRef(base_iri + 'Benign')

code_class = URIRef(base_iri + 'Malware_Code_Structure')

basic_class = URIRef(base_iri + 'Basic')
polymorphic_class = URIRef(base_iri + 'Polymorphic')
metamorphic_class = URIRef(base_iri + 'Metamorphic')
packing_class = URIRef(base_iri + 'Packing')
obfuscation_class = URIRef(base_iri + 'Code-Obfuscation')

family_class = URIRef(base_iri + 'Malware_Family')
```

Fig. 6: Classes of the Ontology

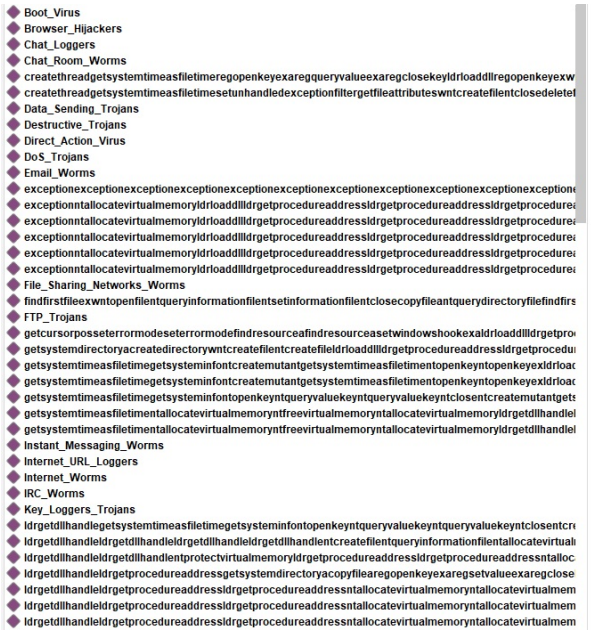

Fig. 7: Individuals of the output ontology

lations for object properties per class.

3) **Inheritance Richness:** Average number of subclasses in a class.
4) **Attribute Richness:** Ratio of the total number of data type properties by the number of classes.
5) **Relationship richness:** 'reflects the diversity of relations', by comparing the number of non-subsumption relations to the number of subsumption relations.
6) **Comprehension of classes:** Percentage of annotation of the classes in ontology.

Metrics of the output ontology were calculated using OntoMetrics[7]. The evaluation statistics of the base metrics of the proposed ontology is shown in Fig. 8. Further statistics such as class axioms, object and data property oxioms can be easily derived with such toolset.

[7]https://ontometrics.informatik.uni-rostock.de/ontologymetrics/

Base metrics more details

Axioms:	557
Logical axioms count:	258
Class count:	29
Total classes count:	29
Object property count:	8
Total object properties count:	8
Data property count:	6
Total data properties count:	6
Properties count:	14
Individual count:	142
Total individuals count:	142
DL expressivity:	ALCR(D)

Fig. 8: Base Metrics the ontology

TABLE I: Evaluation Results

Results	**Output Ontology**	MALOnt	Swimmer's Ontology
Attribute richness	**0.206897**	0.191176	0.0
Class richness	0.310345	**0.970588**	0.0
Inheritance richness	**0.965517**	0.676471	0.941176
Relationship richness	0.333333	**0.402597**	0.0
Average population	**4.896552**	3.897059	0.0

Finally we compare our proposed ontology with state of the art, another two ontologies: MALOnt [18] and Swimmer's Ontology [30]. Results are shown in Table I, showing the terminology box (Tbox) for the comparison of the ontologies. The results shows superiority (in most cases) of the proposed framework. Although it is fairly challenging, such evaluation metrics are useful to quantify various aspects of the ontology frameworks, especially for the comparison purposes.

V. Conclusion

The proposed ontology is an approach to build a knowledge of a malware domain which intends to contribute and complement within cyber-security domain. This ontological structure will be beneficial for knowledge sharing and knowledge management. The proposed ontology includes several contributions including 1) a detailed understanding of different malware families with their behaviour, 2) it captures the changes of API call sequences (behaviour) of metamorphic malware and to comprehend how this type of malware can produce absurd

opcodes. This is an approach to understand and provide a well-structured malware behaviour knowledge.

The base ontology was divided in different instances to classify malware types. 'Malware Family' has a number of 'Individuals' for each sub-classes. This work established an object-oriented approach of ontology building and its handling in a dynamic sense. Further development is expected through inclusions of more number of classes and individuals which in essence would improve the richness of the ontology.

REFERENCES

[1] Maochao Xu, Lei Hua, and Shouhuai Xu, "A vine copula model for predicting the effectiveness of cyber defense early-warning," *Technometrics*, vol. 59, no. 4, pp. 508–520, 2017.

[2] Imtithal A Saeed, Ali Selamat, and Ali MA Abuagoub, "A survey on malware and malware detection systems," *International Journal of Computer Applications*, vol. 67, no. 16, 2013.

[3] André Grégio, Rodrigo Bonacin, Antonio Carlos de Marchi, Olga Fernanda Nabuco, and Paulo Lício de Geus, "An ontology of suspicious software behavior," *Applied Ontology*, vol. 11, no. 1, pp. 29–49, 2016.

[4] André Grégio, Rodrigo Bonacin, Olga Nabuco, Vitor Monte Afonso, Paulo Lício De Geus, and Mario Jino, "Ontology for malware behavior: A core model proposal," in *2014 IEEE 23rd International WETICE Conference*. IEEE, 2014, pp. 453–458.

[5] Dennis Kiwia, Ali Dehghantanha, Kim-Kwang Raymond Choo, and Jim Slaughter, "A cyber kill chain based taxonomy of banking trojans for evolutionary computational intelligence," *Journal of Computational Science*, 11 2017.

[6] Hsien-Der Huang, Tsung-Yen Chuang, Yi-Lang Tsai, and Chang-Shing Lee, "Ontology-based intelligent system for malware behavioral analysis," in *International Conference on Fuzzy Systems*. IEEE, 2010, pp. 1–6.

[7] L. C. Navarro, Alexandre K. W. Navarro, A. Grégio, Anderson Rocha, and R. Dahab, "Leveraging ontologies and machine-learning techniques for malware analysis into android permissions ecosystems," *Comput. Secur.*, vol. 78, pp. 429–453, 2018.

[8] Hsiu-Sen Chiang, Woei-Jiunn Tsaur, et al., "Ontology-based mobile malware behavioral analysis," in *IEEE Second International Conference on Social Computing (SocialCOM 2010)*, 2010, vol. 10.

[9] B. Jasiul, J. Sliwa, K. Gleba, and M. Szpyrka, "Identification of malware activities with rules," *2014 Federated Conference on Computer Science and Information Systems*, pp. 101–110, 2014.

[10] Ruili Zhou, Jianfeng Pan, Xiaobin Tan, and Hongsheng Xi, "Application of clips expert system to malware detection system," in *CIS*, 2008.

[11] Gilberto Fernandes, J. Rodrigues, L. F. Carvalho, J. Al-Muhtadi, and M. L. Proença, "A comprehensive survey on network anomaly detection," *Telecommunication Systems*, vol. 70, pp. 447–489, 2019.

[12] Ori Or-Meir, Nir Nissim, Yuval Elovici, and Lior Rokach, "Dynamic malware analysis in the modern era—a state of the art survey," *ACM Comput. Surv.*, vol. 52, no. 5, Sept. 2019.

[13] Shahid Alam, Issa Traore, and Ibrahim Sogukpinar, "Annotated control flow graph for metamorphic malware detection," *Computer Journal*, vol. 58, 10 2015.

[14] Sujandharan Venkatachalam and Mark Stamp, "Detecting undetectable metamorphic viruses," in *Proceedings of 2011 International Conference on Security & Management (SAM'11)*. Citeseer, 2011, pp. 340–345.

[15] Reza Mirzazadeh, Mohammad Hossein Moattar, and Majid Vafaei Jahan, "Metamorphic malware detection using linear discriminant analysis and graph similarity," in *2015 5th International Conference on Computer and Knowledge Engineering (ICCKE)*. IEEE, 2015, pp. 61–66.

[16] Ferhat Ozgur Catak, Ahmet Faruk Yazı, Ogerta Elezaj, and Javed Ahmed, "Deep learning based sequential model for malware analysis using windows exe api calls," *PeerJ Computer Science*, vol. 6, pp. e285, July 2020.

[17] Tala Tafazzoli and Seyed Hadi Sadjadi, "Malware fuzzy ontology for semantic web," *International Journal of Computer Science and Network Security*, vol. 8, no. 7, pp. 153–161, 2008.

[18] Nidhi Rastogi, "Malont: An ontology for malware threat intelligence," 08 2020.

[19] David A Mundie and David M Mcintire, "An ontology for malware analysis," in *2013 International Conference on Availability, Reliability and Security*. IEEE, 2013, pp. 556–558.

[20] Michael Iannacone, Shawn Bohn, Grant Nakamura, John Gerth, Kelly Huffer, Robert Bridges, Erik Ferragut, and John Goodall, "Developing an ontology for cyber security knowledge graphs," in *Proceedings of the 10th Annual Cyber and Information Security Research Conference*, 2015, pp. 1–4.

[21] F. Abdoli, N. Meibody, and R. Bazoubandi, "An attacks ontology for computer and networks attack," in *SCSS*, 2008.

[22] Jannath Nisha O.S and Mary Saira Bhanu S, "Detection of malicious android applications using ontology-based intelligent model in mobile cloud environment," *Journal of Information Security and Applications*, vol. 58, pp. 102751, 2021.

[23] Ping Wang, Kuo-Ming Chao, Chi-Chun Lo, and Yu-Shih Wang, "Using ontologies to perform threat analysis and develop defensive strategies for mobile security," *Information Technology and Management*, vol. 18, no. 1, pp. 1–25, 2017.

[24] Hsien-De Huang, Giovanni Acampora, Vincenzo Loia, Chang-Shing Lee, and Hung-Yu Kao, "Applying fml and fuzzy ontologies to malware behavioural analysis," in *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011)*. IEEE, 2011, pp. 2018–2025.

[25] L. Obrst, P. Chase, and R. Markeloff, "Developing an ontology of the cyber security domain," in *STIDS*, 2012.

[26] Muzzamil Noor, H. Abbas, and W. B. Shahid, "Countering cyber threats for industrial applications: An automated approach for malware evasion detection and analysis," *J. Netw. Comput. Appl.*, vol. 103, pp. 249–261, 2018.

[27] Weijie Han, "Aptmalinsight: Identify and cognize apt malware based on system call information and ontology knowledge framework," *Information Sciences*, vol. 546, pp. 633–664, 09 2020.

[28] Jean-Baptiste Lamy, "Owlready: Ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies," *Artificial Intelligence in Medicine*, vol. 80, pp. 11–28, 2017.

[29] Ivan Georgiev, Supervisor Eng, and Schaaf, *Cyber Security Fraud Prevention using Data Analytics Developing a Layered Framework with Preconditions to Enable Fraud Identification in Bank Sector*, Ph.D. thesis, 12 2017.

[30] M. Swimmer, "Towards an ontology of malware classes," https://dokumen.tips/documents/towards-an-ontology-of-malware-classes.html, Jan. 2008.