MALOnt: An Ontology for Malware Threat Intelligence

Nidhi Rastogi[1] [0000−0002−2002−3213], Sharmishtha Dutta[1] [0000−0002−4464−1462], Mohammed J. Zaki[1] [0000−0003−4711−0234], Alex Gittens[1] [0000−0003−3482−0157], and Charu Aggarwal[2] [0000−0003−2579−7581]

[1] Rensselaer Polytechnic Institute, Troy, NY, USA 12180
[2] IBM T. J. Watson Research Center,Yorktown Heights, NY, USA 10598
`(raston2,duttas,gittea)@rpi.edu, zaki@cs.rpi.edu, charu@us.ibm.com`

Abstract. Malware threat intelligence uncovers deep information about malware, threat actors, and their tactics, Indicators of Compromise, and vulnerabilities in different platforms from scattered threat sources. This collective information can guide decision making in cyber defense applications utilized by security operation centers. In this paper, we introduce an open-source malware ontology, MALOnt that allows the structured extraction of information and knowledge graph generation, especially for threat intelligence. The knowledge graph that uses MALOnt is instantiated from a corpus comprising hundreds of annotated malware threat reports. The knowledge graph enables the analysis, detection, classification, and attribution of cyber threats caused by malware. We also demonstrate the annotation process using MALOnt on exemplar threat intelligence reports. A work in progress, this research is part of a larger effort towards auto-generation of knowledge graphs for gathering malware threat intelligence from heterogeneous online resources.

Keywords: Malware· Threat Intelligence· Ontology· Knowledge Graphs

1 Introduction

Malware attacks impact every industry that is enabled by Internet technology — approximately 7.2 billion malware attacks were reported worldwide in 2019[3]. Such attacks cause loss, alteration, and misuse of sensitive data and compromise system integrity. Malware often have typical patterns corresponding to the type of industry they attack, groups of attackers behind similar attacks and means to pave their way into the target system, traces left behind after an attack has taken place, and so on. For preventing and detecting future attacks - both similar and disparate, the collection, integration, and analysis of malware threat intelligence is crucial.

A malware ontology can support the construction of models that can detect and track attacks from their initial stages (such as identification of a vulnerability) through later stages (such as an exploit or data compromise). An ontology

[3] https://tinyurl.com/yxs8h6aw

acts as a blueprint of a specific domain, containing key concepts (classes), their properties. Restrictions on classes are defined to limit the scope of the class, which is then inherited by the instances as well. Therefore, it can facilitate the aggregation, representation, and sharing of threat information which would otherwise be challenging to reproduce, reuse, and analyze at a large scale. Both human and software agents can use an ontology to understand the structure of information that is stored in a document, a report, a blog, a tweet, or any other structured, semi-structured, or unstructured information source[11].

Specifically for malware threat, an ontology can provide a dictionary of attacks and related information that can help SOC analysts to dig deeper into their origination, attack goals, timeline, affiliated actors, vulnerabilities exploited for the attack, impact on industries as well as on humans. Such rich pieces of information can be aggregated following the ontology classes and data properties, which can significantly enhance current, future, and sometimes past analysis of online attacks, thereby curbing their propagation before a malware becomes hard to contain. In the absence of an ontology, security researchers and SoC analysts struggle to manage information from multiple sources and rely on ad-hoc mechanisms. There are existing attack and threat taxonomies that can be extended to a knowledge graph, however, they have their limitations that we cover in later sections. Linking common information between threat sources also becomes complex and therefore researchers are either compelled to look at attack instances in isolation or make do with available context.

The main contribution of this paper is MALOnt - an open-source malware ontology[4] which underpins the collection of malware threat intelligence from disparate online sources. MALOnt contains concepts ranging from malware characteristics to attack and attacker details. For example, malware details may include family details, attack vectors (software or hardware vulnerabilities) deployed by an attacker, targeted operating system, impacted industries, history of attacks, and so on. MALOnt can be populated with specific instances and thus, be expanded to generate a knowledge graph (KG). A malware knowledge graph reasoner can infer new facts through deduction or induction and relies on machine learning and deep learning models to greatly expand the range and scale of fact generation. MALOnt will be used as a baseline framework for generating the KG, which is part of our ongoing research. MALOnt is implemented to supplement malware threat information extraction by following the steps below:

1. Create MALOnt - an ontology for malware threat intelligence.
2. Create a malware knowledge graph by integrating malware-related information with the ontology.
3. Infer implicit intelligence from the knowledge graph using an OWL (Web Ontology Language) reasoner.

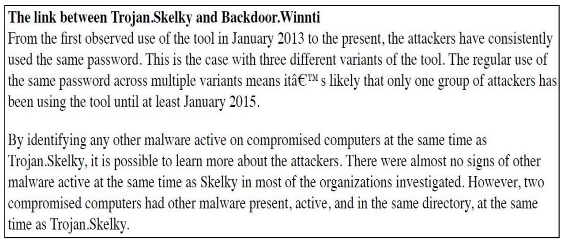
These steps will be described in detail in Section 4.

[4] https://github.com/shoron-dutta/MALOnt

2 Background Concepts

In this section, we cover key concepts that are used to create and instantiate MALOnt.

2.1 Threat Reports



The link between Trojan.Skelky and Backdoor.Winnti
From the first observed use of the tool in January 2013 to the present, the attackers have consistently used the same password. This is the case with three different variants of the tool. The regular use of the same password across multiple variants means itâ€™s likely that only one group of attackers has been using the tool until at least January 2015.

By identifying any other malware active on compromised computers at the same time as Trojan.Skelky, it is possible to learn more about the attackers. There were almost no signs of other malware active at the same time as Skelky in most of the organizations investigated. However, two compromised computers had other malware present, active, and in the same directory, at the same time as Trojan.Skelky.

Fig. 1. An excerpt of a threat report on Backdoor.Winnti malware family.

When a malware attack occurs or when a software vulnerability is identified, a detailed account of these actions is captured by researchers and security analysts in threat reports. Corrective measures are eventually taken to prevent further propagation of the malware, and more evidence is added to the earlier documented accounts. Threat reports are technical in nature and cover the information related to malware (a collective name for viruses, trojan, ransomware, spyware) such as vulnerabilities exploited, operating system and applications impacted, modus operandi, group or cybercriminal responsible, hash of the malware, first sighting of the attack, determined IP addresses of attack server, and so on. Other security analysts and researchers utilize these reports as a reliable source to study and analyze malware samples, adversary techniques, exploited zero-day vulnerabilities, and much more. Figure 1 shows a snippet from a sample threat report [5][6] describing Backdoor.Win32 (Win64) malware family.

2.2 Ontology

An ontology broadly describes concepts within a domain through classes and properties. These properties include property between defined classes and their attributes. An ontology is usually designed around a few main classes that cover the domain whose scope has been predefined. These classes may have sub-classes (more specific), super-classes (more general). The relationship between classes determines the type of interaction between them. Instances are individual instantiated examples of a class, which means each instances can have different values for data properties and be connected to other instances via object properties.

[5] https://tinyurl.com/y9e7m5w7
[6] https://tinyurl.com/y9e7m5w7

Three classes - Malware, Location, and AttackerGroup, largely describe a malware's behavior. This can be vetted with the attack or vulnerability details captured in a few relevant threat reports. The Malware class has two sub-classes - TrojanHorse and Dropper. A property hasTargetLocation exists where the domain is determined by the Malware class and range by the Location class. Two properties exist *from* AttackerGroup class *to* TrojanHorse and Dropper classes titled usesTrojan and usesDropper respectively. A property of the Dropper class is titled deliveredIn that represents the mechanism of how the dropper is delivered to the target system.

To build an ontology, three main approaches are recommended while also engaging human expertise to build them:

1. *Top-down* - Classes are defined from the root of the class hierarchy by identifying the most general classes first [18][5][11]. This approach is preferred when the goal of the ontology is to represent distinctive features of a domain [18].
2. *Bottom-up* - One starts with the leaves in the class hierarchy and builds higher levels of abstraction along the way [18][5][11]. Relevant data sources can be used to identify concepts that are expressed in the dataset.
3. *Middle-out* - The most important classes are determined first followed by the rest of the class hierarchy. It is a combination of top-down and bottom-up approaches [5][11].

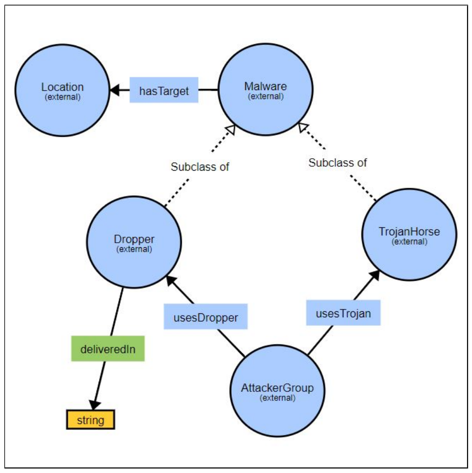


Fig. 2. Visual portrayal of a few top Classes in MALOnt using VOWL plugin in Protege.

Refer to Figure 2 for description on a few of the concepts from MALOnt.

2.3 Knowledge Graphs

A knowledge graph is a machine-readable data repository containing a large amount of structured and unstructured data. This data is stored as triples ¡*Subject, Predicate, Object*¿, where the predicate indicates the relationship between a subject and an object. Each entity or node in a knowledge graph has a unique identifier and may be connected via properties. A unique identifier allows capturing provenance information comprising references to threat sources of the triples. The graph structure can be exploited for efficient information extraction. Ontologies play a crucial role in building knowledge graphs (KGs). One way to build a KG is by adding individual instances to the ontology classes, and properties [11].

In addition to this, class and property instances outside of those defined by the ontology can be added to an existing KG, which allows for flexibility in KG generation. Consider the ontology explained in subsection 2.2. When a small portion of the ontology is populated with instances collected from threat reports, we get a small malware knowledge graph. Such as, the report titled Oops, they did it again: APT Targets Russia and Belarus with ZeroT and PlugX[7]" contains information about an attacker group, which can be mapped to AttackerGroup class. The attacker uses trojans - *PlugX* and *NetTraveler*, to target infrastructures in Europe, Russia, Mongolia, Belarus, among others. This attacker group uses a dropper *Microsoft Compiled HTML Help (.chm)* which is delivered via spear-phishing involving bespoke emails. This information is mapped to MALOnt classes (described in subsection 2.2), to generate a small part of the malware knowledge graph, as visualized in Figure 3. A KG is not just an instantiated data of an ontology. It is a web of properties between individual nodes (also called entities) and uses a reasoner to draw connections between entities that would otherwise not be understood.

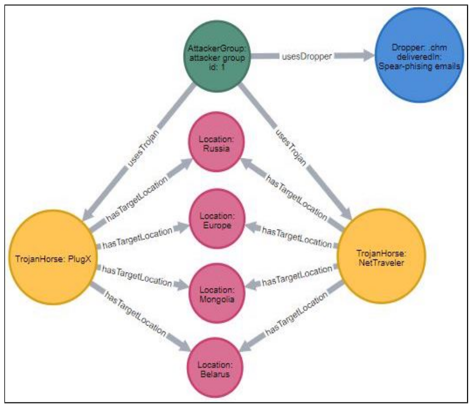
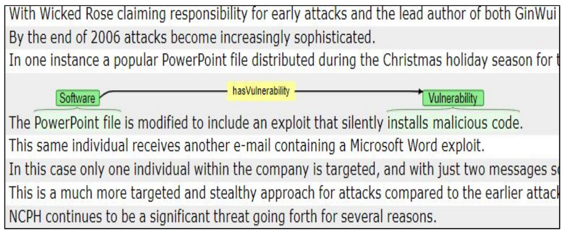


Fig. 3. Snippet of an exemplar malware knowledge graph using neo4j.

[7] https://tinyurl.com/yavqfb2y

Threat Reports are annotated (see Figure 4) using annotation tools such as Brat Rapid Annotation Tool[19], and INCEpTION to create instances of classes defined in MALOnt. Here, the text segments "PowerPoint file" and "installs malicious code" are labeled as MALOnt classes titled Software and Vulnerability respectively. The arrow from Software to Vulnerability denotes the semantic relationship between these two classes hasVulnerability.



With Wicked Rose claiming responsibility for early attacks and the lead author of both GinWui
By the end of 2006 attacks become increasingly sophisticated.
In one instance a popular PowerPoint file distributed during the Christmas holiday season for t
The PowerPoint file is modified to include an exploit that silently installs malicious code.
This same individual receives another e-mail containing a Microsoft Word exploit.
In this case only one individual within the company is targeted, and with just two messages s
This is a much more targeted and stealthy approach for attacks compared to the earlier attack
NCPH continues to be a significant threat going forth for several reasons.

Fig. 4. Annotation using Brat[19].

3 Literature Review

We corroborate the timeliness and necessity of MALOnt in this section by explaining the gaps in existing approaches and by comparing it with the state of the art standards, taxonomies, and ontologies.

3.1 Existing Malware ontologies

Swimmer[21] presented one of the first classifications of malware and their behavior using two classes: *Malware* and *MalwareCharacteristic*. Stucco[6] expressed functionalities for capturing information on an attack although it lacked the means to store properties such as malware type(dropper, trojan, etc.) or attacker's location. Unified Cyber Ontology (UCO)[22] is based on STIX[2] and other cybersecurity standards and is mapped to vocabularies and sharing standards in the cybersecurity domain, as well as external sources such as DBPedia. To the best of our knowledge, the entire UCO is not publicly available and is broader in scope when compared to MALOnt. As normally practiced, some of the cybersecurity concepts for basic classes (e.g. MalwareCharacteristic[21]) were imported, however, MALOnt goes beyond describing just the malware attack behavior. It also captures the impacted industries, malware propagation mechanism, timeline, targeted system, prevention, and so on.

One might argue against the need for a malware ontology since there exist quite a few standards and taxonomies, as well as ontologies in the cybersecurity domain that can be used to share malware threat intelligence in a structured way.

We compare some of the most prominent ones with MALOnt here. Mitre's Common Vulnerabilities and Exposures (CVE)[8] dictionary identifies publicly known security vulnerabilities in software packages. Common Attack Patterns Enumerations and Characteristics (CAPEC)[9] provides an enumeration of repeated techniques in cyber attacks. Mitre's Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK)[10] provides a list of publicly known adversaries, their techniques, and post-compromise tactics to achieve their objectives. OpenIOC[11] is a standard format for sharing IOCs. CVE[9], CAPEC[1], ATT&CK, and IoCs provide static information of already discovered malware artifacts about malware attacks (among other information), which facilitates the representation and integration of collected information. STIX[2], a knowledge representation standard, is expressed in XML which does not support reasoning or identifying properties between class instances.

The aforementioned standards cannot parse multitudes of threat advisory information and present it in a meaningful, human-readable, actionable format that can be used by AI models [17] for prediction or analysis. It is our observation that the earlier work focuses on ontologies from specific threat vectors, such as malware. MALOnt differs from these largely because the domain is beyond threat Due to big data available in the cyber threat landscape, we strive to create information extraction techniques that can perform automated analysis, enable reasoning, enhance inference capabilities with minimal human intervention. This feature is currently lacking in available standards. Therefore, we propose the use of Web Ontology Language or OWL[12] as the language for malware threat knowledge representation and analysis.

3.2 Knowledge Graphs for Malware

Generating knowledge graphs for malware threat intelligence is an emerging research area. This is partly due to a limited background in KG and in adopting its concepts for security research. The paper closest to MALOnt and the proposed malware KG is[14][15], where a pipeline to create knowledge graphs from after action reports (similar to threat reports) is proposed. Existing standards and vocabularies in cybersecurity were used in conjunction with the threat reports to prepare the training dataset for the cybersecurity KG.

In contrast, a combination of vector spaces and a knowledge graph was proposed in [10]. Vector embedding can be more efficient in searching similar nodes whereas knowledge graphs enable reasoning. These complementary strengths were used to build a pipeline comprising knowledge extraction, representation, and querying of data objects which performed better than its components.

Aside from these, multitude domain-specific knowledge graphs have been built around an existing ontology or a generic knowledge base such as WikiData

[8] https://cve.mitre.org/
[9] https://capec.mitre.org/
[10] https://attack.mitre.org/
[11] https://www.fireeye.com/blog/threat-research/2013/10/openioc-basics.html
[12] https://www.w3.org/OWL/

[7][20][23], and made openly accessible to the scientific community. While undoubtedly useful, they cater to generic concepts in the real world such as Person, Organization, Location. To the best of our knowledge, there exists no open-source knowledge graph in the malware threat intelligence domain that captures sufficient details to enable large scale automated malware threat analysis.

4 Ontology Design & Implementation

In this section, we describe the methodology and scope for defining MALOnt classes and properties, the requirement criteria, MALOnt's intended application, as well as example classes and properties.

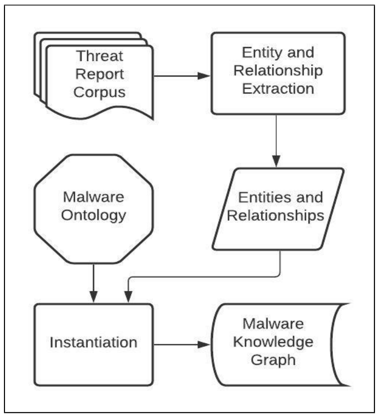


Fig. 5. Using a Malware Ontology to construct a Malware Knowledge Graph

4.1 Purpose and intended use of MALOnt

Malware threat reports are written in natural language and describe malware attacks in detail. Information retrieval from such data feeds can be unstructured in nature, which poses several challenges for information extraction.

An ontology can serve the purpose of mapping disparate data from multiple sources into a shared structure using a common vocabulary that would further facilitate the collection, aggregation, and analysis of that data[13]. Therefore, we propose MALOnt - a malware ontology to encapsulate the concepts necessary to represent information regarding malware. The intended purpose of MALOnt is threefold:

1. To capture semantic information from threat reports by assigning individual entities or instances to a pre-defined class in MALOnt and identifying properties where applicable.

2. To use factual evidence present in the reports and infer new connections and properties between instances.
3. To serve as a foundation for creating a malware KG by populating MALOnt with individual instances from threat reports.

The steps required to achieve these goals are shown in a graphical format in Figure 5. Once instantiated, MALOnt can extract information such as the indicators of compromise, adversary information, software vulnerabilities, attack tactics, and much more. It would also assist researchers and security analysts who gather malware intelligence from unstructured sources. Furthermore, software agents can utilize MALOnt to generate malware KGs.

4.2 Competency Questions for MALOnt

Before creating an ontology, it's requirements [11] should be gathered, defined, and scoped by answering relevant competency questions. Having these competency questions act as the north star when identifying pertinent classes and properties for proper coverage of the domain. SPARQL queries can either be used to answer a question or a narrower version of a broad competency question by running them on the instantiated ontology.

For MALOnt, the domain of the ontology is cybersecurity. In order to create the scope within the larger cybersecurity domain, over two dozen threat reports and existing ontology related sources (owl files, and research papers) were reviewed. Key terms from the reports were identified and the hierarchy of existing ontologies was studied. This helped us vet out other ontologies, import relevant classes to MALOnt, and create the class hierarchy that adequately covers different aspects of malware threat intelligence. Below are the three competency questions that broadly cover the scope of malware threat intelligence:

1. Which malware characteristics adequately define malware threat landscape? (including methods, vulnerabilities, targets, and cybercriminals.)
2. What are the similar features for grouping adversaries, malware to help understand their behavior and predict the future course of action?
3. What is the impact of a given malware on an organization or industry? (financial, human life, intellectual property, reputation)

4.3 Creating MALOnt

Designing and developing an ontology is an agile process. Three stages were continuously visited and reviewed based on core-competency questions - reviewing threat reports, identifying classes, hierarchy, data properties, and evaluating existing security ontologies. The middle-out approach in creating ontology also covers the first two stages. For MALOnt, many top-level classes were created with hierarchy and data-properties abstracted from threat reports. Instances were created for these classes by capturing individuals from threat reports. In the rest of this section, we use examples to describe the middle-out approach for building MALOnt.

The pre-defined upper-level classes such as *Malware*, incorporate some of the most relevant details about a malware attack. Host, Information, MalwareCharacteristics, Malware are extracted from existing ontologies [6][21][3]. The family of a specific malware is represented by *MalwareFamily*. These two classes are joined by the property *hasFamily* where *Malware* class is its domain and *MalwareFamily* class is the range.

Thereafter, we reviewed the competency questions defined in Section 4.2 and identified classes such as *Attacker*, *Organizations*, and *Indicator*. Over two dozen threat reports were manually reviewed to identify key concepts that needed to be included in the ontology as new classes or properties.

For example, threat reports frequently provide valuable details about software vulnerabilities exploited by the malware, as well as specific release or version of the software product. In consequence, we included a *Software* class with two properties *hasReleaseYear* and *hasVersion*. To connect the instance for class *Software* to its vulnerability, *hasVulnerability* property is introduced.

4.4 Exemplar Classes and Relationships

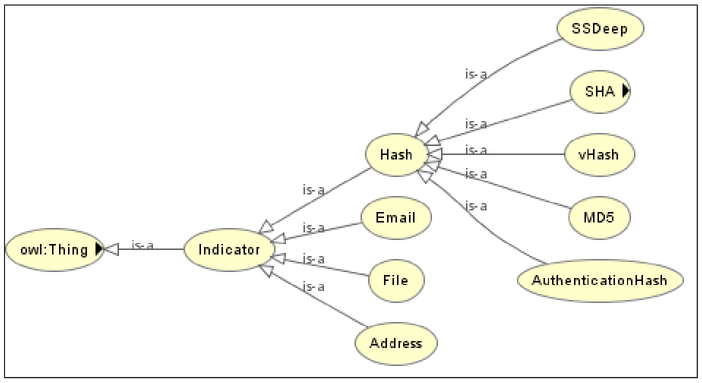


Fig. 6. Indicator class in MALOnt using OWLviz plugin in Protege[12].

In this section, we list out and describe some of the top-level classes and properties in MALOnt, which are essential to extract information from malware threat reports.

- Malware: The general concept of malware, which is malicious software intended to violate the integrity, availability, or confidentiality of a computer system[8]. It has four sub-classes.
- MalwareFamily: A group of malware with common properties. Often threat reports describe the behavior of a malware family (see Figure 1) to help detect or prevent a novel malware belonging to that family.

– Attacker: An adversary or a cybercriminal who can cause damage to a computer system by illegal methods. It is assumed that all attackers in this class are humans.
– AttackerGroup: A group of cybercriminals who have homogeneous signatures of attack.
– ExploitTarget: An entity (a person or an organization) that is the target of a malware attack.
– Indicator: Distinguishable artifacts in a computer system that indicates malicious or suspicious behavior.
– Location: Geographic location of a place.

The Indicator class construct can be seen in Figure 6. Since an indicator of compromise (IoC) can be of different forms (file, email, hash, address), four sub-classes are created to define them. Furthermore, a malware hash signature is of different types, and the six sub-classes of the Hash cover them.

Next, we describe a few properties that represent the semantics of the sentences and connect the instances in malware threat reports. The domain is used to define property characteristics whereas range enforces restrictions. Together they help maintain the integrity of the ontology, contain the possibility of inconsistencies, or erroneous conclusions by the automated reasoners.

– *hasVulnerability*: Bridges an exploit target or a software to its vulnerability.
Domain: *ExploitTarget*, *Software*
Range: *Vulnerability*
– *hasAttachment*: Creates link from a malicious email to the attachment it contains
Domain: *Email*
Range: *File*
– *indicates*: Connects an indicator of compromise to its origin. It has an inverse relation titled *indicatedBy*
Domain: *Indicator*
Range: *Malware*
– *usesDropper*: Connects a malware or an adversary to a frequently used measure - a dropper.
Domain: *Attacker*, *Malware*, *Campaign*
Range: *Dropper*
– *hasFamily*: Maps a malware to its family. There is an inverse relation of this named *hasMember*
Domain: *Malware*
Range: *MalwareFamily*
– *hasCharacteristics*: Maps a malware instance to its behavioral attributes
Domain: *Malware*
Range: *MalwareCharacteristics*

Class specific properties, called datatype properties, are shown in Figure 2. For example, the class Software has two properties - hasVersion and hasReleaseYear.

5 Evaluation

In this section, we evaluate MALOnt by running SPARQL queries on the ontology. These SPARQL queries answer to specific use cases of the competency questions, explained in Section 4.2. This method of evaluation is referred to as goal modeling [4] and is considered a very effective evaluation technique to test the adaptability and consistency of an ontology [16]. If the SPARQL queries are able to extract instances as a response, it signifies that the competency questions have succeeded in covering the defined goal of the ontology.

1. **Retrieving threat information related to malware characteristics.** Competency question 1 can have a specific use case, where MALOnt is queried to extract attributes of different malware campaigns. MALOnt's property targets is selected from Campaign to Organization to get a list of all malware campaigns, their respective target organizations as well as persons (see SPARQL query in Listing1.1).

   Listing 1.1. SPARQL Query for Competency Question 1

```
SELECT DISTINCT ?instance ?p ?o
WHERE {
    ?instance a ?x .
    ?instance ?p ?o .
    ?p a owl:ObjectProperty .
    ?x a owl:Class .
    ?x rdfs:label
        "Campaign"^^xsd:string .
    ?p rdfs:label "targets"
    }
```

2. **Retrieving similar features for grouping concepts**.
   Competency question no.2 can take various forms. Here, we show a SPARQL query that leverages the inverse properties of classes in MALOnt, to find all malware families whose member malware have left a specific IoC footprint. In Figure 7, each instance of Malware class in MALOnt is mapped to an instance that belongs to class MalwareFamily using property titled hasFamily. Alternatively, there is an inverse relation of hasFamily called hasMember. Malware instances can also be mapped to Indicator instances using indicatedBy property. The inverse of this property is indicates.
   Once an IoC (defined by class Indicator) of a given malware is extracted from a threat report, the property is identified, which traces the malware back to the malware family. More insights about a malware family can be gathered through such chain properties between the three MALOnt classes.
   A SPARQL query is executed for detecting instances of *MalwareFamily* that have any member malware indicated by an *Indicator* class. The SPARQL query traverses two kinds of triple structures, ⟨*MalwareFamily, hasMember, Malware*⟩ and ⟨*Malware, indicatedBy, Indicator*⟩ in order to find the response, see listing 1.2:

Listing 1.2. SPARQL Query for Competency Question 2

```
SELECT DISTINCT ?malware_family ?p
                    ?malware ?q ?t
WHERE {
    ?malware_family ?p ?o .

    ?malware_family a ?x.
    ?x a owl:Class.
    ?x rdfs:label "MalwareFamily"
        ^^xsd:string.
    ?p a owl:ObjectProperty.
    ?p rdfs:label "hasMember".

    ?malware ?q ?t .

    ?malware a ?z .
    ?z a owl:Class.
    ?q a owl:ObjectProperty .
    ?q rdfs:label "indicatedBy"
        ^^xsd:string .
    ?t a owl:NamedIndividual .

    ?t rdfs:label "indicator_value"
}
```

3. **Retrieving information on affected person or organization**.
   For competency question no. 3, one can extract information about affected systems, organizations, or person(s).

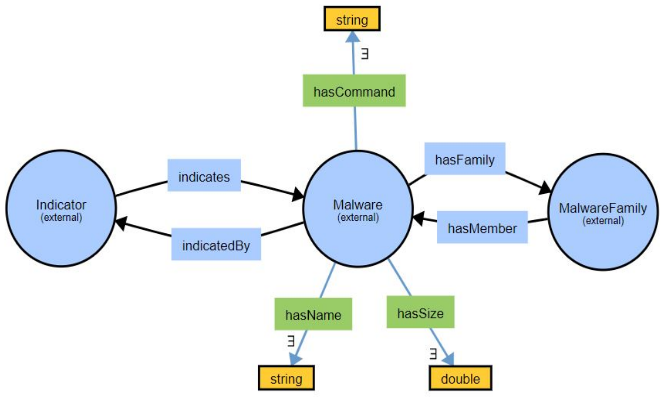


Fig. 7. Inverse properties in MALOnt using VOWL plugin in Protege.

The SPARQL query in listing 1.3 retrieves a list of target objects, and accessed information from those objects by a specific attacker group. This query can retrieve all information of a particular AttackerGroup entity using properties where AttackerGroup is the domain.

Listing 1.3. SPARQL Query for Competency Question 3

```
SELECT DISTINCT ?instance ?p ?o ?q
WHERE {
        ?instance ?p ?o .
        ?instance a ?x .
        ?instance rdfs:label
            "AttackerGroup1"^^xsd:string .
        ?p a owl:ObjectProperty .
        ?x a owl:Class .
        ?p rdfs:label ?q .
        ?o a ?object .
        ?object a owl:Class .
 }
```

6 Application of MALOnt

In this section, we demonstrate the process of creating a part of the KG (due to space constraints) by instantiating MALOnt with over a dozen threat reports. We also explain how the reasoner can be used to retrieve information by capturing it from multiple threat reports by executing SPARQL queries on the exemplar KG.

6.1 Annotating Threat Reports

MALOnt has been instantiated with open-source threat reports [13]that were published between the years 2006 to 2020. Many of these reports have been published by reputed organizations working within the cybersecurity domain. These reports provide a range of coverage on malware threats prominent at the time of publication. A few other reports focus on homogeneous attributes of various attacks caused by malware.

For example, a report [14] published in 2011 covers details on a set of operations known as Night Dragon. Another report[15]published in 2013, focuses on Night Dragon, Stuxnet, and Shamoon. Annotating different kinds of reports in the corpus allows deeper and wider range of details on a particular malware attack.

Threat reports were manually annotated by the authors and reviewed by a security expert. The annotated values from the threat reports were used to instantiate various concepts of MALOnt. In this step, values are assigned to

[13] https://tinyurl.com/y9shcvpd
[14] https://tinyurl.com/y5veq59m
[15] https://tinyurl.com/y52axjtf

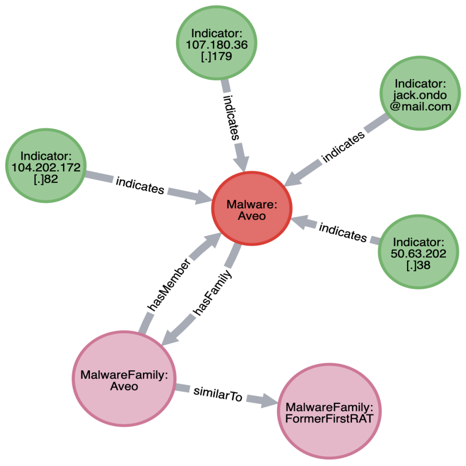

Indicator: 107.180.36 [.]179
Indicator: jack.ondo @mail.com
Indicator: 104.202.172 [.]82
Malware: Aveo
Indicator: 50.63.202 [.]38
indicates
indicates
indicates
indicates
hasMember
hasFamily
MalwareFamily: Aveo
similarTo
MalwareFamily: FormerFirstRAT

Fig. 8. Exemplar knowledge graph based on MALOnt in neo4j

instances of MALOnt classes and properties. In Figure 8 the snippet of the malware KG depicts modeling of threat data collected from reports using classes and properties from the MALOnt ontology.

7 Conclusion and Future Work

In this paper, we propose MALOnt - an ontology for malware threat intelligence by defines 68 classes, 31 properties, and 13 properties for representing malware attacks. We have used the middle-out approach for creating ontologies to review the top classes as well create classes that would be mandatory for a malware threat report. While this is work in progress, we have annotated dozens of threat reports manually to eventually feed the annotations to train predictive named entity recognition models. As future work, MALOnt will further formalize the implicit assumptions of the malware threat domain in order to build a sustainable knowledge graph. For this, annotated malware threat reports will continue to be reviewed and instantiated for MALOnt classes and relations.

8 Acknowledgement

This work is supported by the IBM AI Research Collaboration (AIRC). The authors would like to thank RPI researchers Shruthi Chari and Dr. Oshani Seneviratne for evaluating MALOnt and for ensuring that best practices are followed during ontology generation; Destin Lee for putting together the ontology and knowledge graph figures, and instantiating threat intelligence reports.

References

1. Barnum, S.: Common attack pattern enumeration and classification (capec) schema description. Cigital Inc, http://capec. mitre. org/documents/documentation/-CAPEC_Schema_Descr iption_v1 **3** (2008)
2. Barnum, S.: Standardizing cyber threat intelligence information with the structured threat information expression (stix). Mitre Corporation **11**, 1–22 (2012)
3. Costa, D.L., Albrethsen, M.J., Collins, M.L.: Insider threat indicator ontology. Tech. rep., Carnegie Mellon University Pittsburgh PA United States (2016)
4. Fernandes, P.C.B., Guizzardi, R.S., Guizzardi, G.: Using goal modeling to capture competency questions in ontology-based systems. Journal of Information and Data Management **2**(3), 527–527 (2011)
5. Hendler, J., Ding, Y.: Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool (2012)
6. Iannacone, M., Bohn, S., Nakamura, G., Gerth, J., Huffer, K., Bridges, R., Ferragut, E., Goodall, J.: Developing an ontology for cyber security knowledge graphs. In: Proceedings of the 10th Annual Cyber and Information Security Research Conference. pp. 1–4 (2015)
7. Lockard, C., Dong, X.L., Einolghozati, A., Shiralkar, P.: Ceres: distantly supervised relation extraction from the semi-structured web. Proceedings of the VLDB Endowment **11**(10), 1084–1096 (2018)
8. Mavroeidis, V., Bromander, S.: Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence. In: 2017 European Intelligence and Security Informatics Conference (EISIC). pp. 91–98. IEEE (2017)
9. Mell, P., Scarfone, K., Romanosky, S.: A complete guide to the common vulnerability scoring system version 2.0. In: Published by FIRST-forum of incident response and security teams. vol. 1, p. 23 (2007)
10. Mittal, S., Joshi, A., Finin, T.: Thinking, fast and slow: Combining vector spaces and knowledge graphs. arXiv preprint arXiv:1708.03310 (2017)
11. Noy, N.F., McGuinness, D.L., et al.: Ontology development 101: A guide to creating your first ontology (2001)
12. Noy, N.F., Sintek, M., Decker, S., Crubézy, M., Fergerson, R.W., Musen, M.A.: Creating semantic web contents with protege-2000. IEEE intelligent systems **16**(2), 60–71 (2001)
13. Oltramari, A., Cranor, L.F., Walls, R.J., McDaniel, P.D.: Building an ontology of cyber security. In: Semantic Technology for Intelligence, Defense and Security. pp. 54–61. Citeseer (2014)
14. Pingle, A., Piplai, A., Mittal, S., Joshi, A., Holt, J., Zak, R.: Relext: relation extraction using deep learning approaches for cybersecurity knowledge graph improvement. In: Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. pp. 879–886 (2019)
15. Piplai, A., Mittal, S., Joshi, A., Finin, T., Holt, J., Zak, R.: Creating Cybersecurity Knowledge Graphs from Malware After Action Reports. Tech. rep. (November 2019)
16. Raad, J., Cruz, C.: A Survey on Ontology Evaluation Methods. In: Proceedings of the International Conference on Knowledge Engineering and Ontology Development, part of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management . Lisbonne, Portugal (Nov 2015). https://doi.org/10.5220/0005591001790186, https://hal.archives-ouvertes.fr/hal-01274199

17. Rastogi, N.: A Network Intrusion Detection System (NIDS) Based on Information Centrality to Identify Systemic Cyber Attacks in Large Systems. Ph.D. thesis, Rensselaer Polytechnic Institute (2018)
18. Semy, S., Hetherington-Young, K., Frey, S.: Ontology engineering: An application perspective. In: Wissensmanagement. pp. 499–504 (2005)
19. Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S., Tsujii, J.: Brat: a web-based tool for nlp-assisted text annotation. In: Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics. pp. 102–107. Association for Computational Linguistics (2012)
20. Subasic, P., Yin, H., Lin, X.: Building knowledge base through deep learning relation extraction and wikidata. In: AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering (2019)
21. Swimmer, M.: Towards an ontology of malware classes. Online] January **27** (2008)
22. Syed, Z., Padia, A., Finin, T., Mathews, L., Joshi, A.: Uco: A unified cybersecurity ontology. In: Workshops at the Thirtieth AAAI Conference on Artificial Intelligence (2016)
23. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)