# Semantic model of attacks and vulnerabilities based on CAPEC and CWE dictionaries

Andrei Brazhuk

***Abstract*— This paper discusses the problem of extracting and using knowledge of public directories of software attacks and vulnerabilities to build semantic threat models. The possible purpose of such models is using as a core of a knowledge management system in the software security field. The reason of using the semantic approach (ontologies, reasoning) is a huge number of different data sources in this field and difficulties to analyse them by hand. The proposed semantic model (OWL ontology) is based on the attack pattern (CAPEC) and weakness (CWE) concepts, and can "answer" the questions (by the DL and SPARQL queries), related to grouping (classification) of security concepts according given criteria. The implementation includes free software module (Java, OWL API), able to obtain the OWL ontology from the CAPEC and CWE files in the XML format. To illustrate given ideas, the Protege ontology editor, Pellet reasoner, and SNAP SPARQL plugin are used.**

***Keywords*— attack pattern, CAPEC, CWE, ontology, OWL software security, weakness.**

## I. Introduction

Modern applications should be considered as complex products, which use information technologies itself, telecommunications, cyber-physical technologies, and technologies of the Internet of Things. The challenges of software security require consideration of all the aspects of software design and implementation. To guarantee a particular degree of security a developer should be sure that an application is protected from any potential vulnerability. However, to implement a threat it often needs only one software weakness, which might be converted to a successful attack to the application.

The solving of security issues is very close to the knowledge management. The knowledge management technologies are significant, because of a huge number of different data-sources in the field of software security and difficulties to analyse them by hand. The most suitable knowledge management systems are based on the semantic approach, which uses methods and technologies oriented to the semantic, i.e. reasoning and ontologies [1]. Usually a knowledge management system has an ontology (set of ontologies) as a core. The ontology as a description of terms and relations between them can be used to integrate data or to create a conceptual model of system design. It can also be used as a part of artificial intelligence system or distance-learning system.

Ontology-based systems use the descriptive logics (subset of the first-order logics) as a background. As an advantage of descriptive logic, there is an ability to describe concepts of a subject-specific area and relations between them in very formal way. It is possible to add reasoning with relatively low computational complexity (under certain conditions). For practical tasks one can use OWL (Web Ontology Language), which has initially been created for the Semantic WEB. OWL can also be used for any knowledge-based system.

The current work addresses the problem of extracting and using knowledge of attack and software vulnerabilities directories to build semantic threat models. Such models can be used for a wide range of tasks, related to the synthesis of security systems and security assessment, such as comparing the effectiveness of various security methods and security automation procedures.

Manuscript received January 7, 2019.

Andrei Brazhuk is with the Yanka Kupala State University of Grodno, Grodno, Republic of Belarus (e-mail: brazhuk@grsu.by).

## II. Review the problem

As a part of the software development process, consideration of possible threats for a particular application should be based on experience, guides, and best practices. This should take into account not only positive results, but negative ones too. The most valuable sources of knowledge there are various directories of vulnerabilities, threats, and attacks.

The CWE (Common Weakness Enumeration) and CAPEC (Common Attack Pattern Enumeration and Classification) dictionaries should be considered as universal sources. CWE is a formal list of software weaknesses (types of vulnerabilities). Each CWE record contains a description and set of characteristics, saved in the XML format. CWE records are associated with records of CAPEC or attack patterns. Each attack pattern has a set of formal characteristics for storing records and uses XML too.

To solve the grouping (classification) problem, CAPEC and CWE have own hierarchies of entities. CAPEC uses two basic representations of attack patterns: by mechanism of attack and by domains (target objects) of attacks. CWE operates in three ways: research concepts, development concepts, and architectural concepts. Obviously, to solve the classification problems, existing hierarchies of CAPEC and CWE are not enough. In practice, flexible means of grouping concepts in accordance with current consumer needs are required. Also, the possibility of joint analysis of attack and vulnerability directories should be provided.

There are some scientific researches, related to

development of complex security semantic models in the OWL format. For example, the paper [2] has considered the task of design of ontology-based data model as a part of network attack modelling for the SIEM (Security information and event management) systems. The proposed decisions are based on SCAP (Security Content Automation Protocol). In particular, they describe the common data model, which mentions CAPEC and CWE; also, they depict the ontology of vulnerabilities, based on CVE (Common Vulnerabilities and Exposure).

The work [3] has described UCO (Unified Cybersecurity Ontology), that is aimed to provide information integration and cyber situational awareness in cybersecurity systems. The ontology incorporates huge number of data and knowledge cybersecurity schemas and most commonly used standards for information sharing and exchange (at least they mention CAPEC, CWE and CVE). The best result there might be that UCO is the first cybersecurity ontology, that has been mapped to general world ontologies (DBPedia, Yago). This work is based on STIX (Structured Threat Information eXpression), which is the effort to unify cybersecurity information sharing by incorporating vocabulary from several standards. However, they have used STIX version 1, and that is a challenge to adopt that great work to the more modern STIX version 2.

Also, there are some papers, considering separately the semantic models: only CAPEC [4] or only CWE [5].

The paper [6] has researched a problem of using taxonomies, ontologies and standards for CTI (cyber threat intelligence). They have figured out, none of existed semantic models covers all the information needed to provide effective CTI; and the problem of creation of multi-layered cyber threat intelligence ontology is still actual. That includes description of formal terminology (definitions) and vocabularies, consideration of all the layers of CTI, gathering and formal representation of knowledge, and applying the reasoning capabilities [6].

## III. Structure of Semantic Model

According to the terminology of CWE and CAPEC, the “weakness” and “vulnerability” concepts are different. A *weakness* exists in an application when there is a mistake, which can be related to the architecture, design, coding, or deployment. A *vulnerability* is a weakness that can be used by an attacker to perform destructive actions, causing a negative technical impact to the application or its environment.

The structure of semantic model of software weakness is shown in Fig. 1. The “appearedAt” property shows that a weakness might be added at a particular phase of the software life cycle. CWE and CAPEC both operate the next phases: Architecture and design, Build and compilation, Implementation, Installation, Operation, Requirements, System configuration. The “isDetectedBy” property depicts a possible method of weakness detection. The “DetectionMethod” concept includes a set of common approaches, used to detect weaknesses, i.e. “Black box” or “White box”, “Automated analysis” or “Manual analysis”. For example, “Manual analysis” can be classified as “Manual static analysis” and “Manual dynamic analysis”.

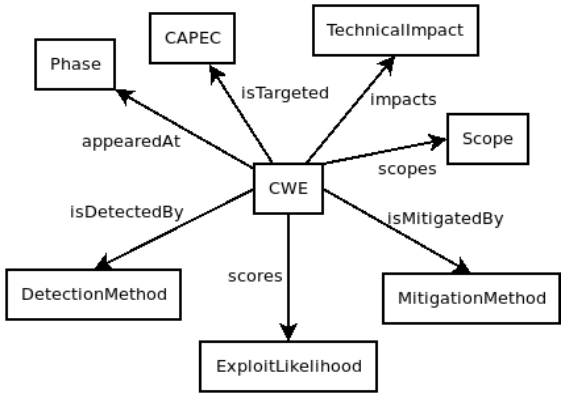


Fig. 1. Structure of semantic model of software weakness

The “isMitigatedBy” property shows a possible method of weakness mitigation. The “MitigationMethod” concept contains different approaches, used to reduce risk, related to a weakness (i.e. “Input validation”, “Sandbox or jail”).

The “scores ExploitLikelihood” property gives a relative quantitative estimation (1-8), describing probability of creation an exploit (sample of destructive code) for a weakness.

The “impacts TechnicalImpact” property maps a weakness concept to a technical impact, which shows effect that the weakness might produce. The “TechnicalImpact” concept includes number of impacts, such as “Alter execution logic”, “Bypass protection mechanisms” etc.

The “scopes” property depicts security properties, violated by a realization of weakness. The “Scope” (or SecurityProperty) concept includes “Confidentiality”, “Integrity”, “Availability”, “Access Control”, “Non repudiation”, “Accountability”, “Authentication”, “Authorization”. The “isTargetedBy” property maps the CWE and CAPEC entities. Originally it refers as “Related weaknesses” or “Related CAPECs”, and we are considering that the next way: if CWE entry refers to CAPEC entries, it might be used by these CAPEC entries and vice versa.

According to the terminology of CWE and CAPEC, the “attack” and “attack pattern” concepts are different. An *attack* is the use of an exploit (a set of exploits) to take advantage of a weakness to obtain a negative technical impact. Also, an attack can be considered as a sequence of steps (sub attacks), i.e. as a complex concept. An *attack pattern* describes common attributes and approaches, utilized by attacker to exploit known weaknesses. The conception of attack patterns comes from the conception of design patterns. The design patterns are applied in constructive context, while the attack patterns are aimed to bring negative context.

The structure of semantic model of attack pattern is shown in Fig. 2. The “scores Likelihood” property gives a relative quantitative estimation (1-8), describing probability of attack success. The “scores Severity” property shows a relative quantitative estimation (1-8), describing severity of attack. The “scores RequiredSkills” property shows a relative quantitative estimation (1-8), describing skills and knowledge, which attacker should have. Obviously, these estimations have been got by methods of expert review.

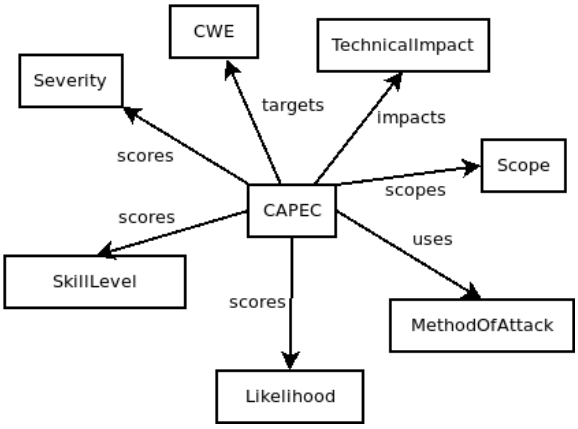


Fig. 2. Structure of semantic model of attack pattern

The "uses MethodOfAttack" property depicts a method, used to perform a particular attack. Attack patterns, as well as weaknesses, are described by the scope and "impacts TechnicalImpact" properties too. Also, the "targets CWE" property (inverse to isTargetedBy) connects the CAPEC concepts to CWE concepts.

## IV. Implementation and Using

The proposed implementation includes two components: the software, able to create semantic model, and the semantic model itself.

The developed Java software module builds the semantic model of attacks and vulnerabilities, based on the XML files. For CAPEC it uses the view of mechanism of attack (https://capec.mitre.org/data/xml/views/1000.xml.zip), for CWE it has to use the research concepts (https://cwe.mitre.org/data/xml/views/1000.xml.zip). To read XML the standard package javax.xml.parsers has been used; it contains API, able to manipulate an object model (DOM - Document Object Model) of XML. The OWL API version 5 external library (https://github.com/owlcs/owlapi) has been used to generate OWL. The Apache Maven (https://maven.apache.org/) is responsible for deployment and building of the software. The source code of the developed software module has freely been published by Github(https://github.com/nets4geeks/abCAPECCWESemanticModel).

The proposed semantic model of attacks and vulnerabilities is represented as an OWL ontology in functional syntax. For the most of top-layer classes (DetectionMethod, MethodOfAttack, MitigationMethod, Phase, Scope, TechnicalImpact) second layer contains instances. For example, the "Availability" concept is an instance of the "Scope" class. However, CAPEC and CWE concepts have classes on second layer for better reasoning. For example, the "iCAPEC_11" attack pattern is an instance of the "CAPEC_11" class, the last is a subclass of the "CAPEC" class.

The proposed semantic model allows providing multi-aspect analysis of attacks and software vulnerabilities. To illustrate this below is shown the ability of model to "answer" the questions, related to grouping of CAPECs and CWEs, considering a particular set of properties and taking into account connections between concepts. The questions can be represented as DL queries as well as SPARQL queries.

To work with ontology the Protege ontology editor (https://protege.stanford.edu/) and Pellet reasoner are used. The possibility to execute DL queries is available through the standard function of Protege (the "DL query" tab). The DL notation is very close to natural language. For example, the request "*Which attack patterns scope the availability property*" is represented as "*CAPEC and scopes value Availability*", and the request "*Which attack patterns scope the availability property and have the severity more than 5*" looks like "*CAPEC and scopes value Availability and scoresSeverity some xsd:integer[> 5]*".

Also, one can create more complex requests, which use CAPEC and CWE together. For example, "*Which weaknesses are targeted by CAPEC 100 and have the exploit likelihood more than 5*" is represented as "*CWE and isTargetedBy value iCAPEC_100 and scoresExploitLikelihood some xsd:integer[> 5]*"

The execution of the last query is shown in Fig. 3. The interface of Protege allows getting results as classes (the "Direct subclasses" option) and instances (the "Instances" option).

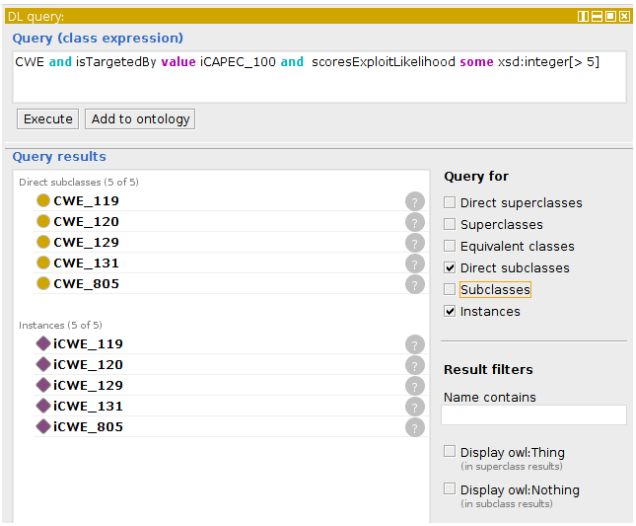


Fig. 3. Execution of DL query by Protege

The DL notation has a restricted set of features, and to perform much more complex requests one might use the SPARQL language. Protege has the SNAP SPARQL plugin, which implements the OWL 2 entailment regime mode [7]. That allows the execution of SPARQL requests on a set of axioms, obtained from inferred ontology (i.e. after a reasoner has processed it).

For example, the request "*Which detection methods of weaknesses might they use to avoid a particular attack pattern (CAPEC 100)?*" using SPARQL might be represented as:

```
SELECT DISTINCT ?x WHERE {
?y :isTargetedBy :iCAPEC_100 .
?y :isDetectedBy ?x .
}
```

The execution of that query by Protege is shown in Fig. 4.

More examples of queries (both DL and SPARQL) one can get from the software module homepage (see above).

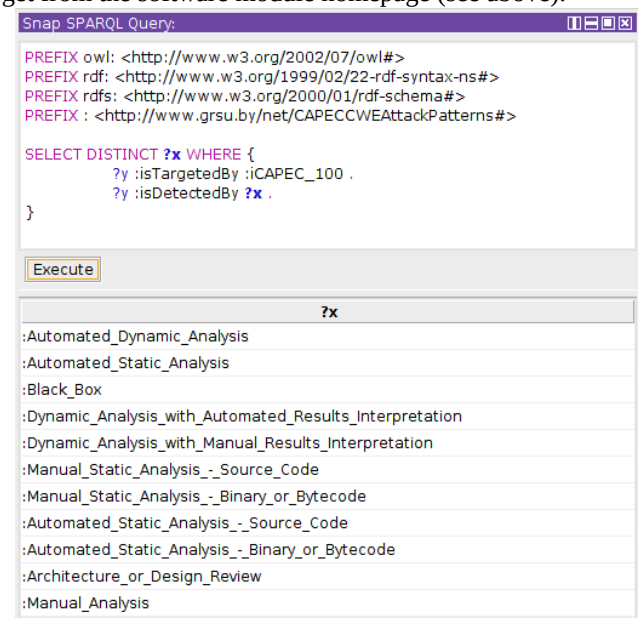


Fig. 4. Execution of SPARQL query by Protege

It should be noted that not all the CAPEC and CWE concepts have a whole description (i.e. contain all the set of properties). Table 1 shows the statistics of used properties by the model.

TABLE I
STATISTICS OF SEMANTIC MODEL

| | **Property** | **Total using** |
|---|---|---|
| 1 | targetsCWE | 871 |
| 2 | impactsTechnicalImpact | 2106 |
| 3 | scopes | 2140 |
| | *CAPEC (512 objects)*: | |
| 4 | usesMethod | 336 |
| 5 | scoresLikelihood | 247 |
| 6 | scoresSeverity | 395 |
| 7 | scoresRequeredSkill | 289 |
| | *CWE (716 objects)*: | |
| 8 | appearedAtPhase | 1126 |
| 9 | isDetectedBy | 411 |
| 10 | isMitigatedBy | 365 |
| 11 | scoresExploitLikelihood | 185 |

## V. CONCLUSIONS

This work shows the process of creation, structure and using of the semantic model of software attacks and vulnerabilities based on the knowledge from public data sources. Here is shown it is possible to obtain the OWL ontology from the CAPEC and CWE dictionaries and use it to answer the questions, represented as the DL or SPARQL queries.

The given model has some limitations. The first one is incompleteness of the model; not all the CAPEC and CWE entries have a full list of properties. The next one is that the formal properties have only been used to depict the attack patterns and weaknesses; the information from description has been omitted, because that would require using of natural language processing methods.

The proposed model “as is” is not so useful, only to show the proof the concept. The possible purpose of the model is using as a core of a knowledge management system. As a part of future research we are going to consider principles, design, and algorithms of a knowledge-based software system, which allows software developers and architects to learn existing software vulnerabilities and attacks, as well as their mitigation and prevention methods. The main challenge here is consideration of security issues in context. If a programmer is writing a PHP-based WEB-application, he would have the most relevant information about how to avoid the vulnerabilities, related to WEB and PHP. If a programmer is creating a low-level network application and using the C++ language, he has to get another piece of security experience from the knowledge-based system. Obviously, these features require using additional data sources, not only CAPEC and CWE, but CVE, which contains lots of details about vulnerabilities of end software products. Also, it needs to apply a dictionary of software products, for example, CPE (Common Platform Enumeration).

## REFERENCES

[1] C. M. Keet, *An Introduction to Ontology Engineering*. 2018.
[2] A. A. Chechulin, I. V. Kotenko, O. V. Polubelova, *Design of the ontology based data model for the network attack modeling system.* Trudy SPIIRAN, 2013, T. 26, pp. 26-39.
[3] Z. Syed, A. Padia, T. Finin, M. L. Mathews and A. Joshi, *UCO: A Unified Cybersecurity Ontology*. In AAAI Workshop: Artificial Intelligence for Cyber Security, 2016.
[4] T. Li, E. Paja, J. Mylopoulos, J. Horkoff, and K. Beckers, *Security attack analysis using attack patterns*. In Research Challenges in Information Science (RCIS), IEEE Tenth International Conference, 2016, pp. 1-13.
[5] Y. Wu, R. Gandhi, and H. Siy, *Using semantic templates to study vulnerabilities recorded in large software repositories.* ICSE Workshop on Software Engineering for Secure Systems, SESS’10, New York, USA, 2010, pp. 22-28.
[6] V. Mavroeidis, S. Bromander, *Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence*, Intelligence and Security Informatics Conference (EISIC), 2017 European, IEEE, 2017, pp. 91-98.
[7] M. Horridge, M. Musen, *Snap-SPARQL: A java framework for working with SPARQL and OWL*. International Experiences and Directions Workshop on OWL, Springer, Cham, 2015. pp. 154-165.

**Andrei Iosifovich BRAZHUK**
Senior lecturer, system administrator at the Yanka Kupala State University of Grodno (https://www.grsu.by)
email: brazhuk@grsu.by
publications: https://scholar.google.com/citations?user=lxR8RLkAAAAJ