An Ontology of Top 25 CWEs

Vladimir Dimitrov and Ivan Kolev

Faculty of Mathematics and Informatics
University of Sofia St. Kliment Ohridki, 5 James Bourchier Blvd., 1164, Sofia, Bulgaria

cht@fmi.uni-sofia.bg, ivan.pl.kolev@gmail.com

Abstract. CWEs Top 25 is a view to the top 25 most dangerous software errors – weaknesses (CWE). The CWE list is maintained by MITRE Corporation. Weaknesses are types of vulnerabilities that can be exploited as vulnerabilities by attacks. MITRE Corporation supports lists for vulnerabilities (CVE) and attacks (CAPEC). The investigation process of given vulnerability, weakness or attack is a sophisticated navigation process in mentioned lists. The aim of presented research is to represent in an ontology Top 25 CWEs and related with them CVEs and CAPECs to facilitate the navigation.

Keywords: weakness, CWE, vulnerability, CVE, attack, CAPEC, ontology, OWL.

1 Introduction

The aim of this research is to be created an ontology in the cybersecurity domain based on information contained in CWE's view Top 25 Most Dangerous Software Errors, and referenced CVEs and CAPECs. Information about CWEs, CVEs and CAPECs is given below in the next section.

The ontology must be simplified to be usable for educational purposes.

2 Basic Terms

The weakness is an error, bug or misconfiguration introduced at some stage of the software life cycle.

The vulnerability is an exploited by some attack weakness. Some weaknesses cannot be exploited because is not available an attack vector to them.

MITRE Corporation maintains:

- CWE [1] – a community developed list of software and hardware weakness types;
- CVE [2] – a list of publicly known cybersecurity vulnerabilities.
- CAPEC [3] – a community developed list of known attack patterns employed to exploit known weaknesses.

Every CVE is linked to concrete vendor(s), product(s) or product version(s).

CWEs are organized in several taxonomies at different abstract levels. CWEs are like CVE types.

MITRE Corporation with above-mentioned lists supports the community process of vulnerability registration, its classification as weaknesses and further investigation of applicable attack patterns. At the beginning is the vulnerability but new weaknesses and attack patterns sometimes have to be introduced.

NIST's NVD [4] is based on CVE. NVD uses the impact metrics CVSS (Common Vulnerability Scoring System) [5] for vulnerability evaluation. CVSS scores of CVEs in NVD are used for ranking Top 25 CWEs.

3 CWE Top 25 Most Dangerous Software Errors

Top 25 Most Dangerous Software Errors (Top 25) [6] is published every year. This is a list of the most widespread and critical weaknesses that can be discovered and exploited as software vulnerabilities.

The ranking methodology intensively uses NVD and CVSS scores assigned there to the CVEs. Only CVEs that have at least one CWE assigned to them as a root cause participate in the calculations.

Two basic values are assigned to each CWE weakness X mentioned in NVD CVEs:

```
frequency – Fr(X)

severity – Sv(X)
```

Let Freq(X) is the number of references to the weakness X in NVD CVEs.

Let Fmin is the minimum value and Fmax is the maximum value of Freq(X) over its domain.

Then weakness X frequency is:

```
Fr(X) = (Freq(X) – Fmin) / (Fmax – Fmin)

Fr(X) value is normalized.
```

Let AVG_CVSS(X) is the average CVSS score from the CVEs in which the weakness X is mentioned as a root cause.

Let CVSSmin is the minimal value and CVSSmax is the maximal value of AVG_CVSS(X) over its domain.

Then the weakness X severity is:

```
Sv(X) = (AVG_CVSS(X) – CVSSmin) / (CVSSmax – CVSSmin)

Sv(X) value is normalized.
```

Finally, the weakness Tip 25 score is:

```
Score(X) = Fr(X) * Sv(X) * 100
```

2019 Top 25 is presented in Table 1. Additionally, CWE team added 15 CWEs (Table 2) that are risky but have not enough score.

Table 1. 2019 Top 25 Most Dangerous Software Errors (source [6]).

Rank	ID	Name	Score
[1]	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	75.56
[2]	CWE-79	Improper Neutralization of Input During Web Page Generation (‘Cross-site Scripting’)	45.69
[3]	CWE-20	Improper Input Validation	43.61
[4]	CWE-200	Information Exposure	32.12
[5]	CWE-125	Out-of-bounds Read	26.53
[6]	CWE-89	Improper Neutralization of Special Elements used in an SQL Command (‘SQL Injection’)	24.54
[7]	CWE-416	Use After Free	17.94
[8]	CWE-190	Integer Overflow or Wraparound	17.35
[9]	CWE-352	Cross-Site Request Forgery (CSRF)	15.54
[10]	CWE-22	Improper Limitation of a Pathname to a Restricted Directory (‘Path Traversal’)	14.10
[11]	CWE-78	Improper Neutralization of Special Elements used in an OS Command (‘OS Command Injection’)	11.47
[12]	CWE-787	Out-of-bounds Write	11.08
[13]	CWE-287	Improper Authentication	10.78
[14]	CWE-476	NULL Pointer Dereference	9.74
[15]	CWE-732	Incorrect Permission Assignment for Critical Resource	6.33
[16]	CWE-434	Unrestricted Upload of File with Dangerous Type	5.50
[17]	CWE-611	Improper Restriction of XML External Entity Reference	5.48
[18]	CWE-94	Improper Control of Generation of Code (‘Code Injection’)	5.36
[19]	CWE-798	Use of Hard-coded Credentials	5.12
[20]	CWE-400	Uncontrolled Resource Consumption	5.04
[21]	CWE-772	Missing Release of Resource after Effective Lifetime	5.04
[22]	CWE-426	Untrusted Search Path	4.40
[23]	CWE-502	Deserialization of Untrusted Data	4.30
[24]	CWE-269	Improper Privilege Management	4.23
[25]	CWE-295	Improper Certificate Validation	4.06

Table 2. Additional 2019 Top 25 Most Dangerous Software Errors (source [6]).

Rank	ID	Name	NVD Count	Avg CVSS
[26]	CWE-835	Loop with Unreachable Exit Condition (‘Infinite Loop’)	218	6.610
[27]	CWE-522	Insufficiently Protected Credentials	150	8.460
[28]	CWE-704	Incorrect Type Conversion or Cast	143	8.484
[29]	CWE-362	Concurrent Execution using Shared Resource with Improper Synchronization (‘Race Condition’)	187	6.740
[30]	CWE-918	Server-Side Request Forgery (SSRF)	128	7.917
[31]	CWE-415	Double Free	111	7.981
[32]	CWE-601	URL Redirection to Untrusted Site (‘Open Redirect’)	159	6.141
[33]	CWE-863	Incorrect Authorization	113	7.050
[34]	CWE-862	Missing Authorization	92	7.491
[35]	CWE-532	Inclusion of Sensitive Information in Log Files	90	7.064
[36]	CWE-306	Missing Authentication for Critical Function	66	8.529
[37]	CWE-384	Session Fixation	76	7.083
[38]	CWE-326	Inadequate Encryption Strength	73	7.278
[39]	CWE-770	Allocation of Resources Without Limits or Throttling	75	6.880
[40]	CWE-617	Reachable Assertion	75	6.729

4 Ontology Contents

CVE, CWE and CAPEC databases are maintained to support the life cycle of vulnerabilities, weaknesses and attack templates. They contain processing information needed for registration, classification and maintenance – not only information related to their nature.

Initially, the new vulnerability is registered in the CVE database by some CVE Numbering Authorities (CNAs). This means that it is yet registered in some private or public repository. Then an investigation process follows to accept or reject this vulnerability. Sometimes, the new vulnerability is recognized as old one. Further, the investigation process relates the vulnerability (CVE entry) to a weakness (CWE entry) and to an attack pattern (CAPEC entry). Sometimes, it is impossible to clarify the vulnerability nature and to relate it with any weakness and/or attack pattern. It is possible a vulnerability to be related to several weaknesses and/or several attack patterns. Briefly,

this is the contents of vulnerability processing without going in details about the life cycle and the phases of vulnerabilities, weaknesses and attack patterns.

In the case of Top 25 Most Dangerous Software Errors, all CVEs are related with CWEs and CAPECs, simply because only such CVEs are used in the ranking procedure.

The ontology is implemented in OWL [7] using Protégé [8]. Initially, it was a master thesis developed by the second co-author under the supervision of the first one. Then it has been redesigned by the first co-author.

CVE, CWE and CAPEC databases are represented in the ontology as disjoint classes. Their definitions are as follow:

```
Class: top25:CAPEC

    DisjointWith: top25:CVE, top25:CWE

Class: top25:CVE

    DisjointWith: top25:CAPEC, top25:CWE

Class: top25:CWE

    DisjointWith: top25:CAPEC, top25:CVE
```

The elements of CVE, CWE and CAPEC have an identifier (ID) – positive integer, a name (Name) – string, and a short description (Description) – string. These characteristics are represented in the ontology as datatype properties:

```
DataProperty: top25:ID

    Characteristics: Functional

    Domain: top25:CAPEC or top25:CVE or top25:CWE

    Range: xsd:string

DataProperty: top25:Name

    Characteristics: Functional

    Domain: top25:CAPEC or top25:CVE or top25:CWE

    Range: xsd:string

DataProperty: top25:Description

    Characteristics: Functional

    Domain: top25:CAPEC or top25:CVE or top25:CWE

    Range: xsd:string
```

CVE entries have external references to other repositories in which they are registered with different identifications. These references are important because they extend the view to the vulnerability but at this time, they are not included in the ontology.

The information about the CVE entry nature is in its description. This is semi-structured text about the vendor, product, version, component root cause, attack vector

etc., but this text is hardly readable even by a human-expert. Information extraction from the CVE description is out of the scope of this research.

CVEs are not organized in any taxonomies – they are simply lists. The information about the CVE entry is contained in its CWE "type", but it is possible a vulnerability to be related with several CWEs. CWEs and CAPECs are organized in several taxonomies.

A CWE entry can have an extended description (ExtendedDescription) in addition to its description. This additional description is included in the ontology as a datatype property:

```
DataProperty: top25:ExtendedDescription

    Characteristics: Functional

    Domain: top25:CWE

    Range: xsd:string
```

Likelihood of CWE entry exploit is evaluated in a scale of several string values. It is included in the ontology as LikelihoofOfExploit datatype property:

```
DataProperty: top25:LikelihoodOfExploit

    Characteristics: Functional

    Domain: top25:CWE

    Range: xsd:string
```

How a CWE can be detected is given in the datatype property DetectionMethods of the class CWE:

```
DataProperty: top25:DetectionMethods

    Domain: top25:CWE

    Range: xsd:string
```

Another important information about the CWE are mitigation methods. These are represented in the datatype property PotentialMitigations:

```
DataProperty: top25:PotentialMitigations

    Domain: top25:CWE

    Range: xsd:string
```

Some CWEs are related to specific programming languages. In the CWE class, this is represented as a datatype property (Languages). This property is a simplification because relation can be not only to the programming languages but also to platforms, technologies etc.:

```
DataProperty: top25:Languages

    Domain: top25:CWE

    Range: xsd:string
```

Usually, the most detailed CWEs (at abstraction level Variant) are linked with some

programming language, platform or technology etc., but in Top 25 Most Dangerous Software Errors, there are CWEs at different abstraction levels. This is another simplification accepting that all CWEs are at the same abstraction level. Just a same is the situation with CAPECs.

CWE entries have more characteristics that are interesting but at this stage only listed above are included in the ontology.

CAPECs are organized in several taxonomies. CAPEC entry has many interesting characteristics, but in our ontology, they participate only with their ID, Name and Description.

One of the most important concept in the ontologies are class relationships. They are modelled as class object properties.

As it has been mentioned, vulnerability "types" are the weaknesses. The object property WeaknessEnumerations links CVEs with their CWEs. The MITRE Corporation CVE list does not contains this relationship, but it is available in NVD with this name. This object property is defined as follows:

```
ObjectProperty: top25:WeaknessEnumerations

    Characteristics: Irreflexive, Asymmetric

    Domain: top25:CVE

    Range: top25:CWE

    InverseOf: top25:ObservedExamples
```

The object property ObservedExamples links weaknesses to their vulnerabilities:

```
ObjectProperty: top25:ObservedExamples

    Characteristics: Irreflexive, Asymmetric

    Domain: top25:CWE

    Range: top25:CVE

    InverseOf: top25:WeaknessEnumerations
```

Weaknesses, on the other hand, are linked with the attack patterns that can be used to exploit them. This relationship is represented as an object property with the name AttackPatterns that links CWEs to their CAPECs:

```
ObjectProperty: top25:AttackPatterns

    Characteristics: Irreflexive, Asymmetric

    Domain: top25:CWE

    Range: top25:CAPEC

    InverseOf: top25:RelatedWeaknesses
```

Finally, the relationship of attack patterns to the exploited by them weaknesses is represented by the object property RelatedWeaknesses:

```
ObjectProperty: top25:RelatedWeaknesses
```

```
Characteristics: Irreflexive, Asymmetric
Domain: top25:CAPEC
Range: top25:CWE
InverseOf: top25:AttackPatterns
```

CVEs and CAPECs are not linked in the ontology and the databases. This relationship can be derived via CWEs.

4 Ontology Usability

Navigation in our ontology is via SPARQL queries. Starting point can be any weakness, vulnerability or attack pattern. The path and its scope depend of the case study scenario. Several case studies, roles (security manager, cyber security operational team, procurement employee, cyber security trainee) and scenarios have been investigated. Identified case studies, roles and scenarios are not exhausting, but it is clear that not all of these potential users can use SPARQL queries in their everyday duties. A specialized user-friendly interface to the ontology must be developed for them.

Presented here ontology is very simple. It contains the basic knowledge about Top 25 Most Dangerous Software Errors and related vulnerabilities and attack patterns but even now, it is populated with 493 individuals (CWEs – 25, CVEs – 156, CAPECs – 312). Further extensions of this ontology would be populated with many thousands CWEs, CVEs and CAPECs. This process have to automatic for a stable ontology structure. CVE, CWE and CAPEC databases are relatively stable but the devil is in the details. So, may be some automatic ontology structure must be developed.

Finally, it is clear that the real power of the semantic search can be achieved with the introduction of CWE and CAPEC taxonomies. This task is not so simple because the used taxonomies are not simple ones and further research and investigations must be done on them.

This research will be used in the context of updating of current curricula at University of Sofia with cybersecurity topics as described in [9].

5 Acknowledgements

This work was conducted using the Protégé resource, which is supported by grant GM10331601 from the National Institute of General Medical Sciences of the United States National Institutes of Health.

This research is supported by the National Scientific Program “Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)”, financed by the Ministry of Education and Science.

References

1. MITRE Corporation, Common Weakness Enumeration, http://cwe.mitre.org, last accessed 2020.
2. MITRE Corporation, Common Vulnerabilities and Exposures, http://cve.mitre.org, last accessed 2020.
3. MITRE Corporation, Common Attack Pattern Enumeration and Classification, http://capec.mitre.org, last accessed 2020.
4. NIST, National Vulnerability Database, http://nvd.nist.gov, last accessed 2020.
5. NIST, Vulnerability Metrics, http://nvd.nist.gov/vuln-metrics/cvss, last accessed 2020.
6. MITRE Corporation, http://cwe.mitre.org/top25, last accessed 2020.
7. W3C, Semantic Web, Web Ontology Language (OWL), http://www.w3.org/OWL, last accessed 2020.
8. Musen, M.A. The Protégé project: A look back and a look forward. AI Matters. Association of Computing Machinery Specific Interest Group in Artificial Intelligence, 1(4), June 2015. DOI: 10.1145/2557001.25757003.
9. Kaloyanova K., Exploring Cybersecurity Curricula Designation Requirements. Computer and Communications Engineering, Vol. 13, No. 2/2019, pp 64-67.