ГОДИШНИК НА СОФИЙСКИЯ УНИВЕРСИТЕТ „СВ. КЛИМЕНТ ОХРИДСКИ“
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА
Том 108

ANNUAL OF SOFIA UNIVERSITY “ST. KLIMENT OHRIDSKI”
FACULTY OF MATHEMATICS AND INFORMATICS
Volume 108

SEMANTIC WEB ECOSYSTEM BASED ON CVE (NVD, CPE), CWE AND CAPEC

VLADIMIR DIMITROV

CVE (NVD, CPE), CWE and CAPEC are databases in the Cybersecurity area sponsored and maintained by the US government. These are lists (databases) organized in taxonomies where it is appropriate. They contain information about known vulnerabilities, weaknesses and attacks. CVE (NVD, CPE), CWE and CAPEC are the corner stone in many cybersecurity tools.
The usage of traditional database systems for the tasks in the cybersecurity require extended knowledge and skills in querying for identification of vulnerabilities, weaknesses and attacks. CVE (NVD, CPE), CWE and CAPEC contain hidden facts and relationships (knowledge) buried in the data. This knowledge can be effectively accessed by the Semantic web tools.
The paper presents an approach for transition to the Semantic web of above-mentioned databases. The approach is presented in illustrative way. This means without duplication with information about the contents available for CVE (NVD, CPE), CWE and CAPEC.

Keywords: cybersecurity, semantic web, CVE, NVD, CPE, CWE, CAPEC

CCS Concepts:
• Security and privacy;
• Computing methodologies~Artificial intelligence~Knowledge representation and reasoning~Ontology engineering

1. CVE, CWE and CAPEC – short presentation

1.1. CVE

CVE (Common Vulnerabilities and Exposures) List [6,7] is maintained by The MITRE Corporation (MITRE). It is sponsored by the U.S. Department of Homeland Security (DHS) Cybersecurity and Infrastructure Security Agency (CISA).

DOI: 10.60063/GSU.FMI.108.29-45

The CVE Project intention is to collect all available vulnerabilities and exposures in a common list to strength cybersecurity worldwide. The CVE List is an effort of the CVE Community in the project.

CVEs are used as a common reference vulnerabilities list in many cybersecurity tools.

Every CVE has a number, a description and source references. The other information is administrative by its nature – there is a process for acceptance of new vulnerabilities.

NVD (National Vulnerability Database) [11] is an extraction from CVE enriched with information from NIST.

CVEs in NVD contain in addition:

- analysis description;
- severity calculated in CVSS Version 3.x and CVSS Version 2.0 [4];
- hyperlinks to advisories, solutions, and tools;
- weakness enumeration – references to CWE weaknesses [8];
- known affected software configurations represented in CPE 2.3 and CPE 2.2 [12].

NVD CVE contains 199 048 CVEs on November 3, 2022.

1.2. CWE

CWE (Common Weakness Enumeration) [8] is sponsored by the DHS Cybersecurity and Infrastructure Security Agency (CISA) and managed by the Homeland Security Systems Engineering and Development Institute (HSSEDI), which is operated by MITRE.

Vulnerabilities are exploitable. They are exposed in certain software and/or hardware configurations. These are weaknesses available for exploitation via some attack vector.

Weaknesses are like vulnerability types. Some vulnerabilities are based on several weaknesses.

The CWE catalog consists of several taxonomies (views). Every view is devoted to a specific audience.

Views can be structured in categories, pillars, classes, bases, and variants, but it is not necessary – they can be simply lists.

Categories contain weaknesses with common characteristics. They are conceptual containers, but not weaknesses.

Pillars, classes, bases, and variants are weaknesses at different abstraction levels. For further information, refer to [8].

It is not necessary the weakness to be exploitable. If an attack vector is not available, the weakness cannot be exploited.

A weakness can participate in several views and categories.

Every weakness contains information about the time of its introduction, detection methods, common consequences of its exploit, how it can be mitigated, affected resources from the exploit, and some other administrative information.

Bases and variants can contain example code as observed examples. They are more specific to technology or language level.

The most important links are to CVEs as observed examples and CAPECs as related attack patterns.

CWE contains 933 CWEs on November 3, 2022.

1.3. CAPEC

CAPEC (Common Attack Pattern Enumeration and Classification) [9] is sponsored by the DHS, CISA and managed by the HSSEDI, which is operated by MITRE. It is publicly available catalog of common attack patterns.

MITRE operate with CWE and CAPEC in the same way as CVE – via open community.

The CAPEC catalog consists of views as CWE. Views can be structured in categories, meta attack patterns, standard attack patterns and detailed attack patterns.

Categories have the same function as in CWE.

Attack patterns can be at different abstraction levels as weaknesses in CWE. They contain information about the prerequisites, i. e. preconditions attack to happen.

CAPECs contain information how to mitigate from the attack.

Some attack patterns contain execution flow structured in: explore, experiment, and exploit.

Attack patterns contain links to CWEs that exploit.

CAPEC contains 555 CAPECs on November 3, 2022.

1.4. The formal knowledge in CVE, CWE, and CAPEC

All above-mentioned databases are under continual improvement and development. They are freely available.

As databases, they can be used in the cybersecurity processes and tools. More specifically, these databases can be used in cybersecurity research and forensics.

CVE is simply a list. There are no classifications for CVEs. Querying CVE list can be performed by CVE ID or by some keyword from the CVE description.

CWEs are organized in several taxonomies. Querying CWE can be performed following some taxonomy hierarchy (view), or as this can be done in CVEs. CWE views are structured or are simply lists.

It is possible in CWE, the same weakness to participate more than once in the same view. For example, see "CWE-653: Improper Isolation or Compartmentalization" that is a child of "CWE-693: Protection Mechanism Failure" and "CWE-657: Violation of Secure Design Principles" in the view "CWE-1000: Research Concepts".

In CAPEC, the situation is just a same as that in CWE.

The intention of the first phase in the development process of CVE, CWE and CAPEC was initially lists of vulnerabilities, weaknesses, and attack patterns to be created. Then, the second phase consists of appropriate structuring to be applied based on some taxonomies. This phase is under development.

The functionality of CVE, CWE and CAPEC, as mentioned above, fully can be implemented with some conventional (relational) database system.

Now, let us go to the ontologies. First, some basic terms have to be introduced.

"By **information** we mean data that have been shaped into a form that is meaningful and useful to human beings. **Data**, in contrast, are streams of raw facts representing events occurring in organizations or the physical environment before they have been organized and arranged into a form that people can understand and use." [5]. In our case, this information is suitable for decision-making support in cybersecurity.

In [1], the terms are more clarified (bolding is by me):

- "**Data** are defined as symbols that represent properties of objects, events and their environment. They are the products of observation. However, are of no use until they are in a useable (i. e. relevant) form. The difference between data and information is functional, not structural.

- **Information** is contained in descriptions, answers to questions that begin with such words as who, what, when and how many. Information systems generate, store, retrieve and process data. Information is inferred from data.

- **Knowledge** is know-how, and is what makes possible the transformation of information into instructions. Knowledge can be obtained either by transmission from another who has it, by instruction, or by extracting it from experience.

- **Intelligence** is the ability to increase efficiency.

- **Wisdom** is the ability to increase effectiveness. Wisdom adds value, which requires the mental function that we call judgement. The ethical and aesthetic values that this implies are inherent to the actor and are unique and personal."

CVE, CWE and CAPEC contain information. CWE and CAPEC contain instructions how to mitigate some weaknesses and attack patterns. Therefore, CWE and CAPEC can be viewed as knowledge bases in the broad meaning of the term. Relationships among CVE, CWE and CAPEC organize an ecosystem in a knowledge base.

Explicit knowledge in these databases is presented as elements and attributes with enumerated values.

The other source of knowledge are the references in the ecosystem. There are references outside the ecosystem – in the Web, but they hardly can be proceeded as links in the ontology sense.

Most of the elements that have text as value are interpreted as annotations in the Semantic web. An exception from this rule is the text value represent atomic fact as knowledge.

In the next sections, a detailed analyze of the explicit knowledge of the targeted databases is performed.

1.5. Knowledge in CVE (NVD)

CVEs contain knowledge in its *description* as unstructured text. There are some recommendations on how vulnerability have to be described in a structured way. However, these rules have not been strictly applied. Because of that, attempts to annotate CVE descriptions as natural text are not successful enough to produce acceptable results.

The CVEs *basic metrics*, *CVSS Version 3.x* and *CVSS Version 2.0*, are codified expert knowledge.

The CVEs are linked to the vulnerable CPEs *configurations* applicable for them. These *configurations* are formally structured. They represent formally codified knowledge about the platforms (software and hardware) and their configurations.

Knowledge about the CVEs is extended to the Web via *external references.* These external links are useful as information source but cannot be proceeded as ontology links.

The CVE's administrative information is not accounted as explicit knowledge here. It is suitable for Semantic web annotation.

Finally, the CVE knowledge is represented by its *description* (unstructured text), *references to vulnerable configurations* (CPEs), *evaluations* (CVSS – formally structured), and via *references* outside the ecosystem.

1.6. Knowledge in CWE

CWEs can be classified in taxonomies (*views*, *pillars*, and *categories*). It is an explicit knowledge presentation.

The *description* and the *extended description* are informal text that cannot be effectively annotated. They are more suitable to be Semantic web annotations.

The *related weaknesses* organize weaknesses in *compound* ones by *nature*, *reference to CWE*, *reference to view*, and possible *reference to chain* and *ordinality.* This is codified knowledge.

The weakness *ordinality* is annotated. It represents the relations of this weakness to the others as *indirect*, *primary*, or *resultant.* This is very vogue knowledge about the weakness, but can be classified as explicit knowledge.

The *applicable platforms* are highly structured by *languages*, *operating systems*, *architectures* and *technologies.* The last ones are codified by *name*, *class* and *prevalence.* In addition, *operating systems* have *version* and *reference to CPE.* This is explicit knowledge.

The *background details* is a simply unstructured text – suitable only for Semantic web annotation.

The *alternative terms* have descriptions. These terms are like nicknames for the weakness and can be accepted as codification (explicit knowledge) of the weakness. The *descriptions of the terms* are annotations to them.

The *modes of introduction*, *applicable platforms*, *common consequences*, *likelihood of exploit*, and *weakness ordinalities* are codified expert knowledge.

The *exploitation factors* is unstructured text suitable only for Semantic web annotations.

The *likelihood of exploit* is an enumeration therefore codified knowledge.

The *common consequences* are codified by scope, impact, and likelihood. They can be annotated, too. This is explicit knowledge with annotations.

The *detection methods* are identified, and codified (explicit knowledge) by *method* and *effectiveness* enumerations. The last two can be annotated.

The *potential mitigations* are represented as semi-structured text. *Mitigations* are structured by *phases*, but the text is not structured and cannot be effectively annotated. Structuring by *phases* is a codified knowledge (explicit knowledge), but the text has to be further structured (codified) to be represented as knowledge.

The *demonstrative examples* are descriptive text and/or links to CVEs. The last one, as relations, is a codified expert knowledge, but the text cannot be effectively annotated and it is suitable to be Semantic web annotation.

The *observed examples* in the best case are references to CVEs, but can be simply text references with *descriptions* suitable for Semantic web annotations.

The *functional areas* and affected resources are enumerations – therefore coded (explicit) knowledge.

The *taxonomy mappings* map to other taxonomies not in the ecosystem. They are formalized (explicit knowledge) by *taxonomy name*, *entry id*, *entry name* and *mapping fit*. These are potential references to other external ontologies.

The *related attack pattern* are references to CAPEC ontology individuals. This is explicit knowledge.

The *references* are references to outside sources – in the best case in the Web. These *references* can be formalized as knowledge if the outside sources are formally represented but it is not the case.

Most of the knowledge about the weaknesses is inside the ecosystem represented via *references*, codified knowledge, structured and semi-structured text.

There are *notes* and *content history* that can be viewed as annotations in the Semantic web.

1.7. Knowledge in CAPEC

CAPEC structure is very similar to that of CWE. The text for the similar elements is copied from CWE to represent fully CAPEC structure.

The *description* and *extended description* are informal text that cannot be effectively annotated. They are more suitable to be Semantic web annotations.

The *alternative terms* have descriptions. These terms are like nicknames for the attack pattern and can be accepted as codification (explicit knowledge) of the attack pattern. *Descriptions* of the terms are annotations to them.

The *likelihood of attack* is an enumeration. Therefore explicit knowledge.

The *common consequences* are codified (explicit knowledge) by *scope*, *impact*, and *likelihood.* They can be annotated too.

The *detection methods* are *identified* and then codified (explicit knowledge) by *method* and *effectiveness* enumerations. The last two can be annotated.

The *potential mitigations* are *identified* and structured (explicit knowledge) by *phases*, *strategies* and *effectiveness* enumerations. The *mitigation* can be annotated. The *mitigation description* as a text is mandatory.

The *demonstrative examples* are descriptive text and/or external references. The last one, as relations, is a codified expert knowledge but the text cannot be effectively annotated and it is suitable to be Semantic web annotation.

The *observed examples*, in the best case, are references to CVEs (explicit knowledge), but can be simply text references with descriptions suitable for Semantic web annotations.

The *functional areas* and *affected resources* are enumerations – therefore coded knowledge.

The *taxonomy mappings* map to other taxonomies not in the ecosystem. They are formalized (explicit knowledge) by *taxonomy name*, *entry id*, *entry name*, and *mapping fit.* These are potential references to other external ontologies.

The *related attack patterns* link the attack pattern with other attack patterns that exploit the same weakness. This is explicit knowledge.

The *references* are references to outside sources – in the best case in the Web. They refer to other terms in outside classification systems. These references can be formalized as knowledge if the outside taxonomies are formally represented but it is not the case.

Most of the knowledge about the attack patterns is inside the ecosystem represented via references, codified knowledge, structured, and semi-structured text.

There are *notes* and *content history* that can be viewed as annotations in the Semantic web.

1.8. Databases vs ontologies

The main problem with these databases is the implicit information. Databases are suitable for "closed world assumption", i. e. all unknown information is false, but this is not the case in cybersecurity research and forensics. In these cases, more suitable is the "open world assumption". The last one is supported in the Semantic web [13].

The main motivation to transfer the knowledge in all these databases into ontologies are the above-mentioned considerations. The ontologies can be used in reasoners to classify automatically the new information, i. e. to create new knowledge for it. In that way, ontologies are not static like the databases – they can be extended with new facts and relationships generated by the classifiers.

2. Ontologies

All ontologies CPE, CVE (NVD), CWE and CAPEC are generated with an option fresh copy from the corresponding site to be downloaded.

The generators are written in Python, which is very useful with its rich libraries and parallelism. They are available from GitHub.

Manchester syntax of OWL [14] has been used. It is easier to develop new ontologies in this syntax – it is human-oriented and supported by Protégé [10].

2.1. CPE ontology

CPE ontology is an ontology that consists of configurations, i. e. its individuals are configurations.

This is very simple ontology supporting CVE (NVD) ontology. It is based on CPE Official Dictionary [12] from NVD.

CPE ontology: classes, object properties and data properties are presented in Figure 1. Here, CPEEntry is disjoint union of CPE and Deprecated or disjoint union of Application, Hardware, OS, and NotAHO.

Deprecation is disjoint union of its subclasses.

Figure 2 represents an individual. The last one use as IRI, CPE 2.2 that is subset of IRI.

Detailed presentation of this ontology and its generator is given in [2, 3].

2.2. CVE ontology

The CVE ontology is based on the NIST NVD – not on the MITRE CVE. This ontology is very huge.

First, every CVE is bounded at least to one vulnerable configuration. To specify these vulnerable configurations, NVD uses CPE Applicability Language. The last

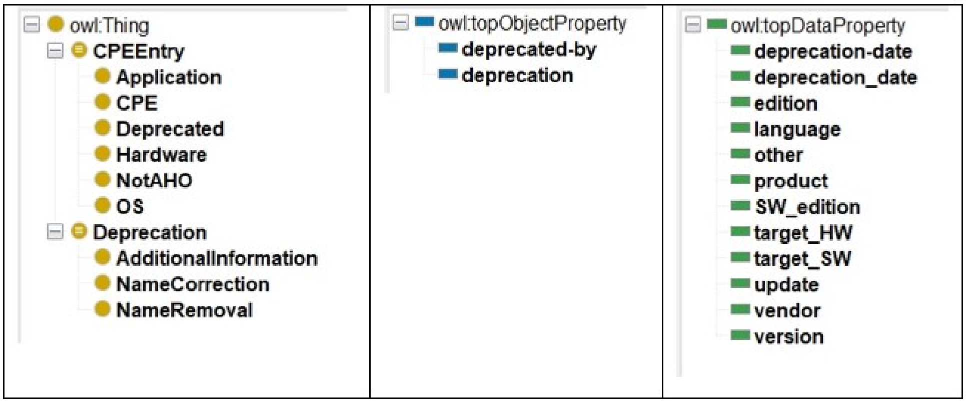
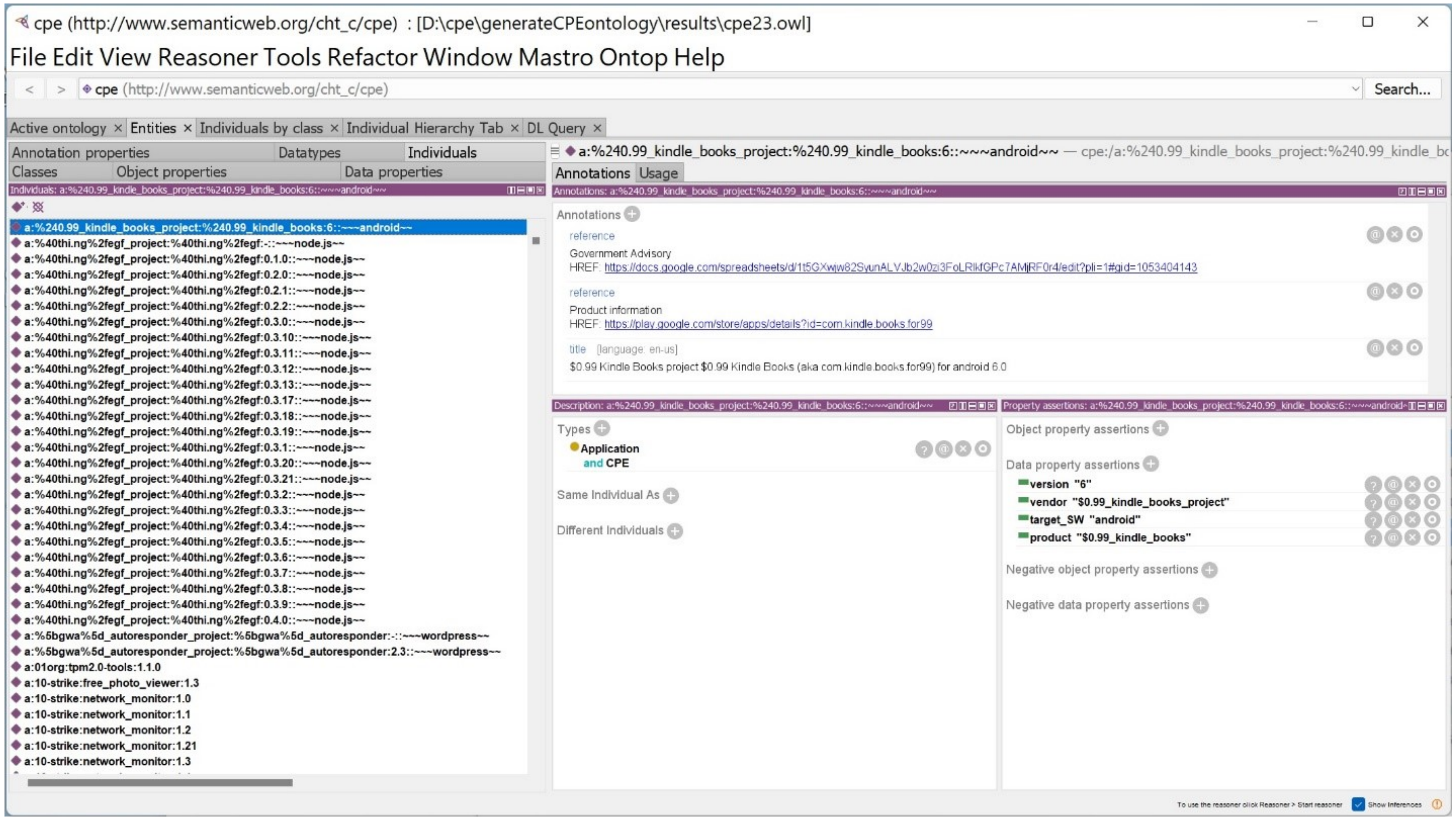


Figure 1. CPE classes and properties


cpe (http://www.semanticweb.org/cht_c/cpe) : [D:\cpe\generateCPEontology\results\cpe23.owl]
File Edit View Reasoner Tools Refactor Window Mastro Ontop Help
cpe (http://www.semanticweb.org/cht_c/cpe)
Search...
Active ontology × Entities × Individuals by class × Individual Hierarchy Tab × DL Query ×
Annotation properties Datatypes Individuals
Classes Object properties Data properties
Annotations Usage
To use the reasoner click Reasoner > Start reasoner
Show Inferences

Individuals: a:%240.99_kindle_books_project:%240.99_kindle_books:6::~~~android~~

```
a:%240.99_kindle_books_project:%240.99_kindle_books:6::~~~android~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.1.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.1::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.2::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.10::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.11::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.12::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.13::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.17::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.18::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.19::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.1::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.21::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.2::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.3::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.4::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.5::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.6::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.7::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.8::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.9::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.4.0::~~~node.js~~
a:%5bgwa%5d_autoresponder_project:%5bgwa%5d_autoresponder:-::~~~wordpress~~
a:%5bgwa%5d_autoresponder_project:%5bgwa%5d_autoresponder:2.3::~~~wordpress~~
a:01org:tpm2.0-tools:1.1.0
a:10-strike:free_photo_viewer:1.3
a:10-strike:network_monitor:1.0
a:10-strike:network_monitor:1.1
a:10-strike:network_monitor:1.2
a:10-strike:network_monitor:1.21
a:10-strike:network_monitor:1.3
```

a:%240.99_kindle_books_project:%240.99_kindle_books:6::~~~android~~ — cpe:/a:%240.99_kindle_books_project:%240.99_kindle_bc

Annotations: a:%240.99_kindle_books_project:%240.99_kindle_books:6::~~~android~~

Annotations

reference
Government Advisory
HREF: https://docs.google.com/spreadsheets/d/1t5GXwjw82SyunALVJb2w0zi3FoLRIkfGPc7AMjRF0r4/edit?pli=1#gid=1053404143

reference
Product information
HREF: https://play.google.com/store/apps/details?id=com.kindle.books.for99

title [language: en-us]
$0.99 Kindle Books project $0.99 Kindle Books (aka com.kindle.books.for99) for android 6.0

Description: a:%240.99_kindle_books_project:%240.99_kindle_books:6::~~~android~~

Types
Application
and CPE

Same Individual As

Different Individuals

Property assertions: a:%240.99_kindle_books_project:%240.99_kindle_books:6::~~~android~

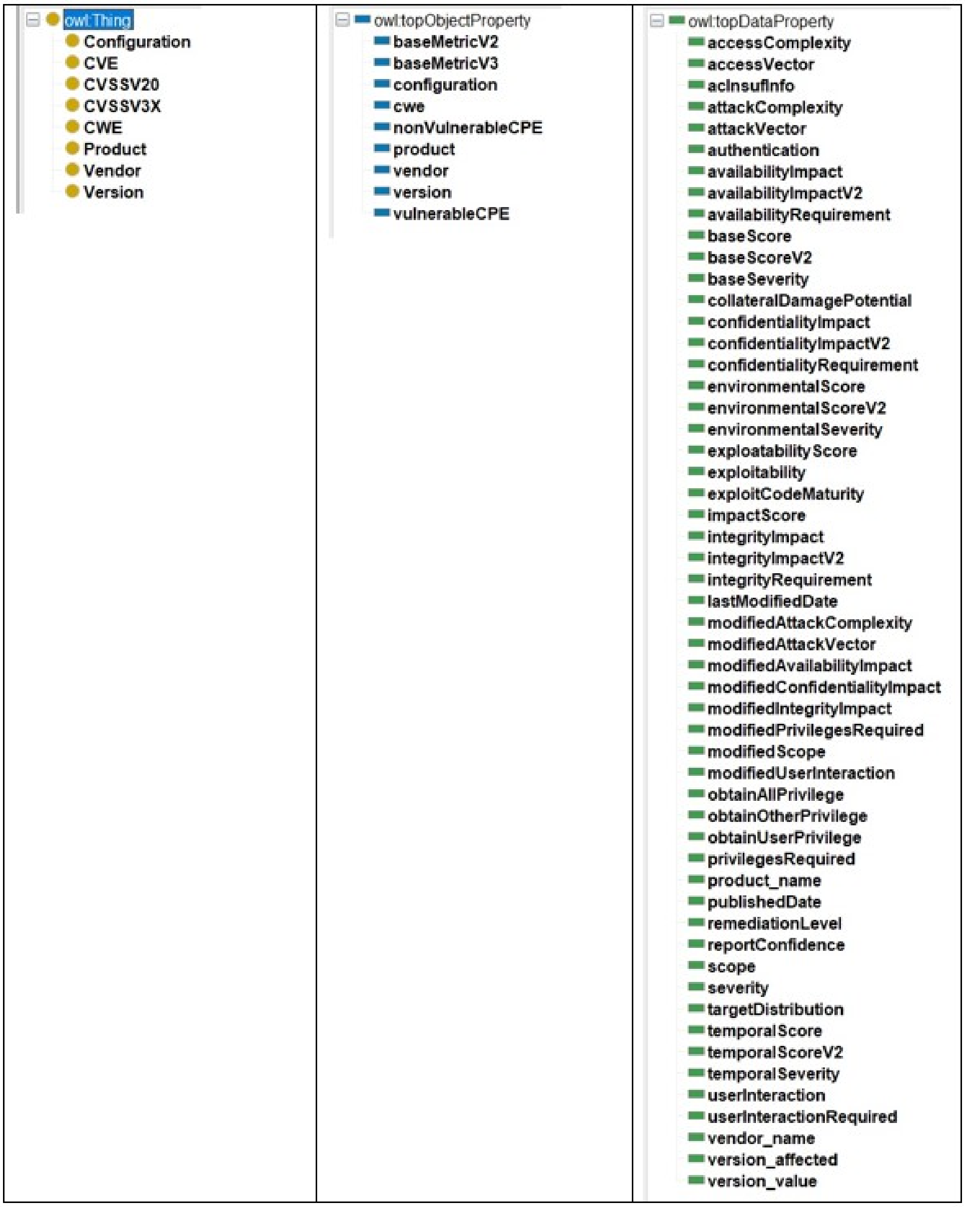
Object property assertions

Data property assertions
version "6"
vendor "$0.99_kindle_books_project"
target_SW "android"
product "$0.99_kindle_books"

Negative object property assertions

Negative data property assertions

Figure 2. CPE individuals



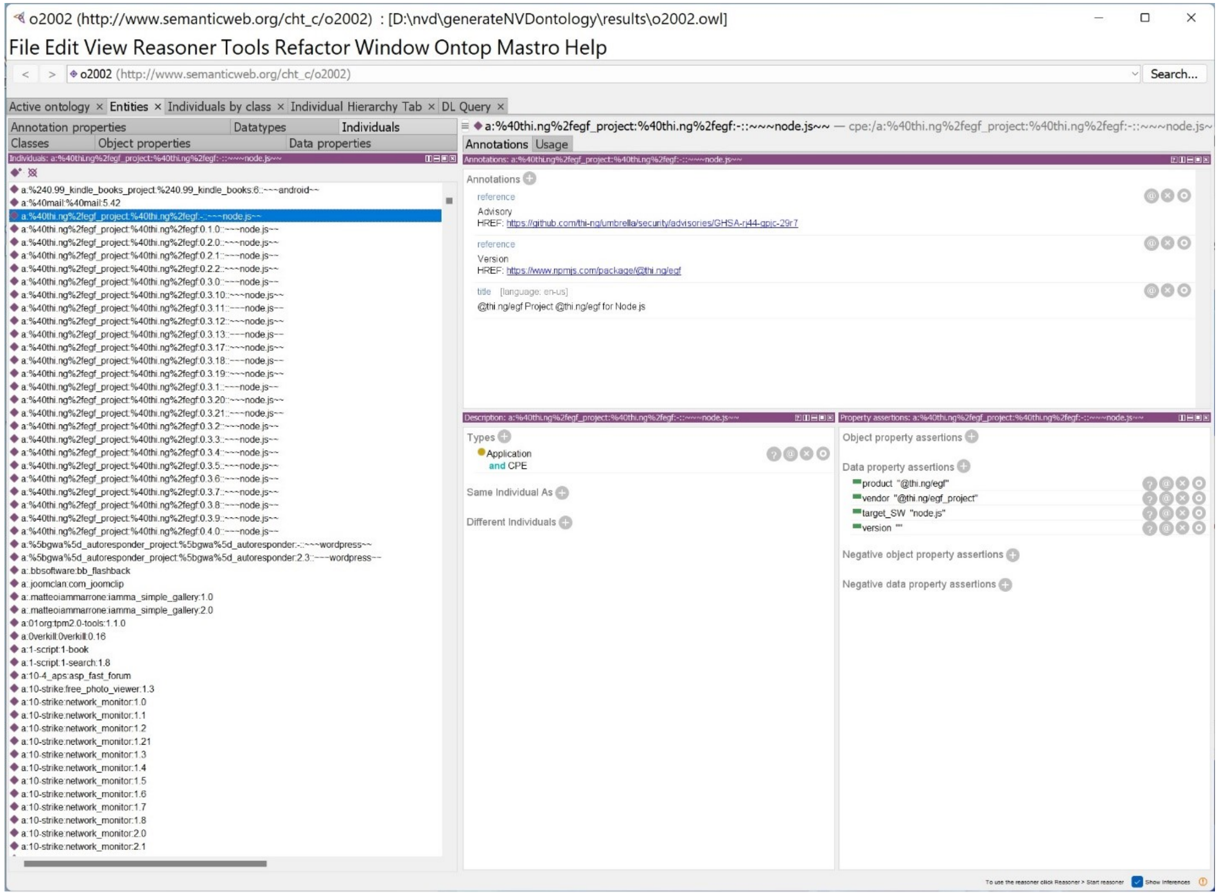
owl:Thing
- Configuration
- CVE
- CVSSV20
- CVSSV3X
- CWE
- Product
- Vendor
- Version

owl:topObjectProperty
- baseMetricV2
- baseMetricV3
- configuration
- cwe
- nonVulnerableCPE
- product
- vendor
- version
- vulnerableCPE

owl:topDataProperty
- accessComplexity
- accessVector
- acInsufInfo
- attackComplexity
- attackVector
- authentication
- availabilityImpact
- availabilityImpactV2
- availabilityRequirement
- baseScore
- baseScoreV2
- baseSeverity
- collateralDamagePotential
- confidentialityImpact
- confidentialityImpactV2
- confidentialityRequirement
- environmentalScore
- environmentalScoreV2
- environmentalSeverity
- exploatabilityScore
- exploitability
- exploitCodeMaturity
- impactScore
- integrityImpact
- integrityImpactV2
- integrityRequirement
- lastModifiedDate
- modifiedAttackComplexity
- modifiedAttackVector
- modifiedAvailabilityImpact
- modifiedConfidentialityImpact
- modifiedIntegrityImpact
- modifiedPrivilegesRequired
- modifiedScope
- modifiedUserInteraction
- obtainAllPrivilege
- obtainOtherPrivilege
- obtainUserPrivilege
- privilegesRequired
- product_name
- publishedDate
- remediationLevel
- reportConfidence
- scope
- severity
- targetDistribution
- temporalScore
- temporalScoreV2
- temporalSeverity
- userInteraction
- userInteractionRequired
- vendor_name
- version_affected
- version_value

Figure 3. CVE classes, object and data properties

one is a kind of query language on the CPE Official Dictionary. However, the reasoners do not execute any kind of queries to extend the ontology. Therefore, these CPE Applicability Language queries have to be extended to “base” configurations from the dictionary.



o2002 (http://www.semanticweb.org/cht_c/o2002) : [D:\nvd\generateNVDontology\results\o2002.owl]

File Edit View Reasoner Tools Refactor Window Ontop Mastro Help
o2002 (http://www.semanticweb.org/cht_c/o2002)
Search...
Active ontology × Entities × Individuals by class × Individual Hierarchy Tab × DL Query ×
Annotation properties Datatypes Individuals
Classes Object properties Data properties

Individuals: a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~~

```
a:%240.99_kindle_books_project:%240.99_kindle_books:6::~~~android~~
a:%40mail:%40mail:5.42
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.1.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.1::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.2::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.10::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.11::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.12::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.13::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.17::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.18::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.19::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.1::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.21::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.2::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.3::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.4::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.5::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.6::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.7::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.8::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.9::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.4.0::~~~node.js~~
a:%5bgwa%5d_autoresponder_project:%5bgwa%5d_autoresponder:-::~~~wordpress~~
a:%5bgwa%5d_autoresponder_project:%5bgwa%5d_autoresponder:2.3::~~~wordpress~~
a:.bbsoftware:bb_flashback
a:.joomclan:com_joomclip
a:.matteoiammarrone:iamma_simple_gallery:1.0
a:.matteoiammarrone:iamma_simple_gallery:2.0
a:01org:tpm2.0-tools:1.1.0
a:0verkill:0verkill:0.16
a:1-script:1-book
a:1-script:1-search:1.8
a:10-4_aps:asp_fast_forum
a:10-strike:free_photo_viewer:1.3
a:10-strike:network_monitor:1.0
a:10-strike:network_monitor:1.1
a:10-strike:network_monitor:1.2
a:10-strike:network_monitor:1.21
a:10-strike:network_monitor:1.3
a:10-strike:network_monitor:1.4
a:10-strike:network_monitor:1.5
a:10-strike:network_monitor:1.6
a:10-strike:network_monitor:1.7
a:10-strike:network_monitor:1.8
a:10-strike:network_monitor:2.0
a:10-strike:network_monitor:2.1
```

a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~~ — cpe:/a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~

Annotations: a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~~

Annotations

reference
Advisory
HREF: https://github.com/thi-ng/umbrella/security/advisories/GHSA-rj44-gpjc-29r7

reference
Version
HREF: https://www.npmjs.com/package/@thi.ng/egf

title [language: en-us]
@thi.ng/egf Project @thi.ng/egf for Node.js

Description: a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~~

Types
Application
and CPE

Same Individual As

Different Individuals

Property assertions: a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~~

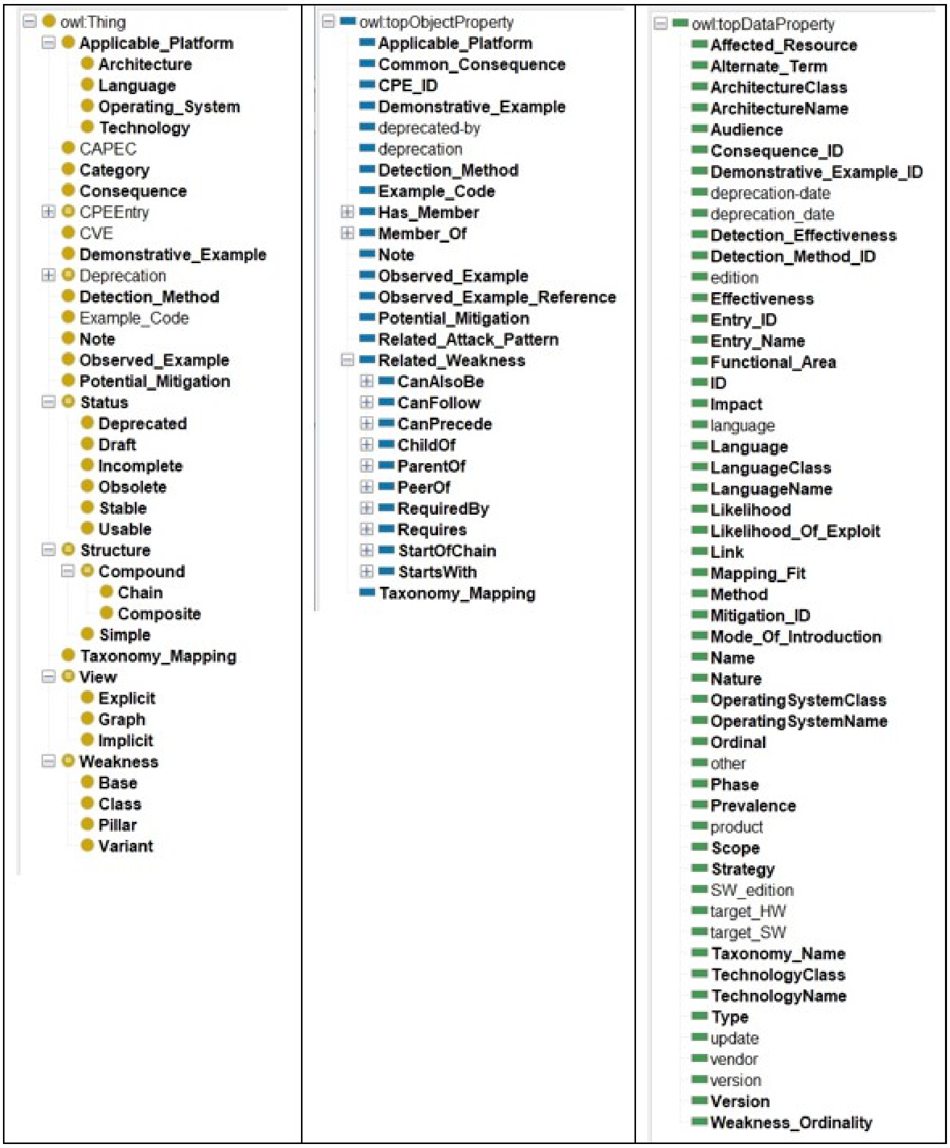
Object property assertions

Data property assertions
product "@thi.ng/egf"
vendor "@thi.ng/egf_project"
target_SW "node.js"
version ""

Negative object property assertions

Negative data property assertions

Figure 4. CVE individuals



- owl:Thing
  - Applicable_Platform
    - Architecture
    - Language
    - Operating_System
    - Technology
  - CAPEC
  - Category
  - Consequence
  - CPEEntry
  - CVE
  - Demonstrative_Example
  - Deprecation
  - Detection_Method
  - Example_Code
  - Note
  - Observed_Example
  - Potential_Mitigation
  - Status
    - Deprecated
    - Draft
    - Incomplete
    - Obsolete
    - Stable
    - Usable
  - Structure
    - Compound
      - Chain
      - Composite
    - Simple
  - Taxonomy_Mapping
  - View
    - Explicit
    - Graph
    - Implicit
  - Weakness
    - Base
    - Class
    - Pillar
    - Variant

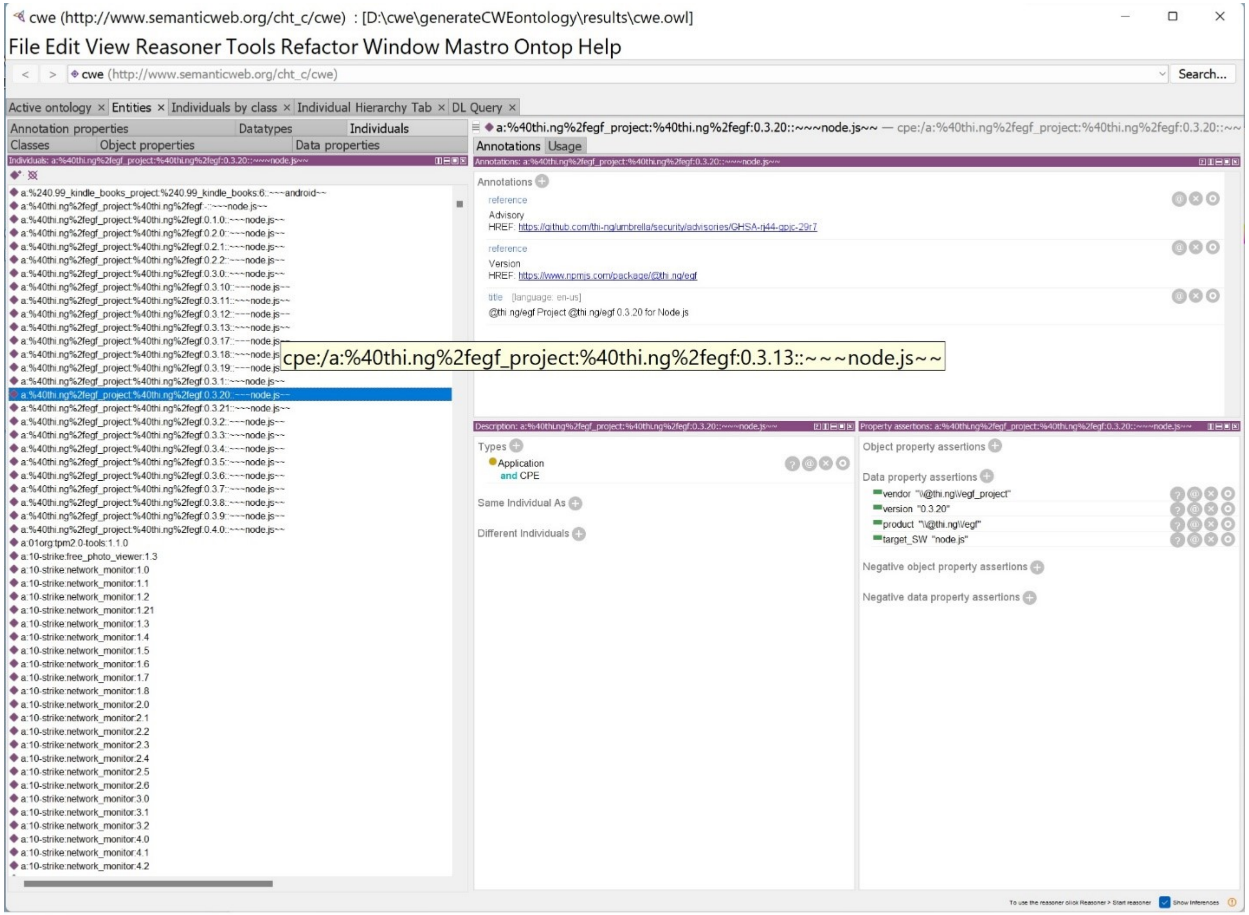
- owl:topObjectProperty
  - Applicable_Platform
  - Common_Consequence
  - CPE_ID
  - Demonstrative_Example
  - deprecated-by
  - deprecation
  - Detection_Method
  - Example_Code
  - Has_Member
  - Member_Of
  - Note
  - Observed_Example
  - Observed_Example_Reference
  - Potential_Mitigation
  - Related_Attack_Pattern
  - Related_Weakness
    - CanAlsoBe
    - CanFollow
    - CanPrecede
    - ChildOf
    - ParentOf
    - PeerOf
    - RequiredBy
    - Requires
    - StartOfChain
    - StartsWith
  - Taxonomy_Mapping

- owl:topDataProperty
  - Affected_Resource
  - Alternate_Term
  - ArchitectureClass
  - ArchitectureName
  - Audience
  - Consequence_ID
  - Demonstrative_Example_ID
  - deprecation-date
  - deprecation_date
  - Detection_Effectiveness
  - Detection_Method_ID
  - edition
  - Effectiveness
  - Entry_ID
  - Entry_Name
  - Functional_Area
  - ID
  - Impact
  - language
  - Language
  - LanguageClass
  - LanguageName
  - Likelihood
  - Likelihood_Of_Exploit
  - Link
  - Mapping_Fit
  - Method
  - Mitigation_ID
  - Mode_Of_Introduction
  - Name
  - Nature
  - OperatingSystemClass
  - OperatingSystemName
  - Ordinal
  - other
  - Phase
  - Prevalence
  - product
  - Scope
  - Strategy
  - SW_edition
  - target_HW
  - target_SW
  - Taxonomy_Name
  - TechnologyClass
  - TechnologyName
  - Type
  - update
  - vendor
  - version
  - Version
  - Weakness_Ordinality

Figure 5. CWE classes, object and data properties

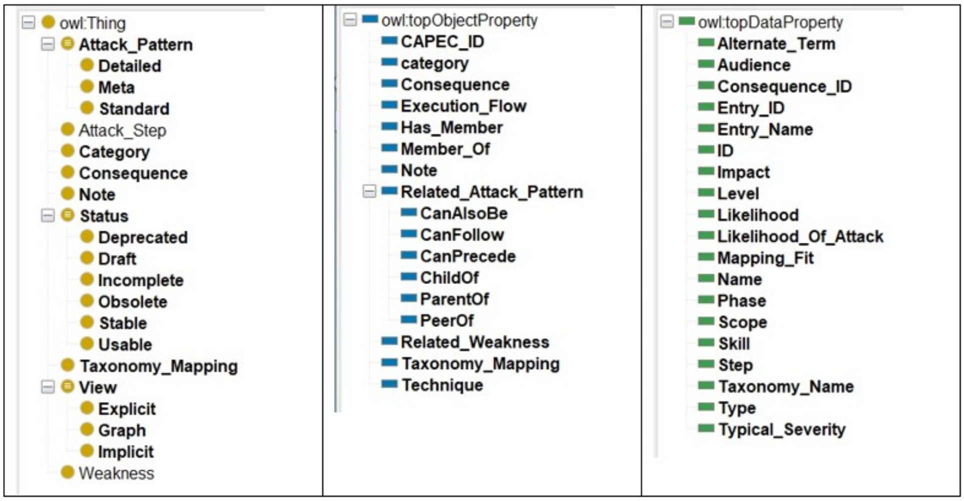
Second, NVD uses so-called CPE Feeds that have to be extensions of the above-mentioned queries to "base" configurations. However, many CPEs from "base" configurations of CPE Feeds are not included in the CPE Official Dictionary. Therefore, for these absent configurations CPE individuals have to be generated.

One more deviation in CPE Feeds to CPE Official Dictionary is that in the last one some CPE templates are not fully extended.



```
cwe (http://www.semanticweb.org/cht_c/cwe) : [D:\cwe\generateCWEontology\results\cwe.owl]
File Edit View Reasoner Tools Refactor Window Mastro Ontop Help
cwe (http://www.semanticweb.org/cht_c/cwe)
Active ontology × Entities × Individuals by class × Individual Hierarchy Tab × DL Query ×

Individuals: a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~~node.js~~
a:%240.99_kindle_books_project:%240.99_kindle_books:6::~~~android~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:-::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.1.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.1::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.2.2::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.0::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.10::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.11::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.12::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.13::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.17::~~~node.js
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.18::~~~node.js
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.19::~~~node.js
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.1::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.21::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.2::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.3::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.4::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.5::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.6::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.7::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.8::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.9::~~~node.js~~
a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.4.0::~~~node.js~~
a:01org:tpm2.0-tools:1.1.0
a:10-strike:free_photo_viewer:1.3
a:10-strike:network_monitor:1.0
a:10-strike:network_monitor:1.1
a:10-strike:network_monitor:1.2
a:10-strike:network_monitor:1.21
a:10-strike:network_monitor:1.3
a:10-strike:network_monitor:1.4
a:10-strike:network_monitor:1.5
a:10-strike:network_monitor:1.6
a:10-strike:network_monitor:1.7
a:10-strike:network_monitor:1.8
a:10-strike:network_monitor:2.0
a:10-strike:network_monitor:2.1
a:10-strike:network_monitor:2.2
a:10-strike:network_monitor:2.3
a:10-strike:network_monitor:2.4
a:10-strike:network_monitor:2.5
a:10-strike:network_monitor:2.6
a:10-strike:network_monitor:3.0
a:10-strike:network_monitor:3.1
a:10-strike:network_monitor:3.2
a:10-strike:network_monitor:4.0
a:10-strike:network_monitor:4.1
a:10-strike:network_monitor:4.2

cpe:/a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.13::~~~node.js~~

a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~~node.js~~ — cpe:/a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~

Annotations: a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~~node.js~~
Annotations
  reference
  Advisory
  HREF: https://github.com/thi-ng/umbrella/security/advisories/GHSA-rj44-gpjc-29r7
  reference
  Version
  HREF: https://www.npmjs.com/package/@thi.ng/egf
  title [language: en-us]
  @thi.ng/egf Project @thi.ng/egf 0.3.20 for Node.js

Description: a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~~node.js~~
Types
  Application
  and CPE
Same Individual As
Different Individuals

Property assertions: a:%40thi.ng%2fegf_project:%40thi.ng%2fegf:0.3.20::~~~node.js~~
Object property assertions
Data property assertions
  vendor "\\@thi.ng\\egf_project"
  version "0.3.20"
  product "\\@thi.ng\\egf"
  target_SW "node.js"
Negative object property assertions
Negative data property assertions
```

Figure 6. CWE individuals



- owl:Thing
  - Attack_Pattern
    - Detailed
    - Meta
    - Standard
  - Attack_Step
  - Category
  - Consequence
  - Note
  - Status
    - Deprecated
    - Draft
    - Incomplete
    - Obsolete
    - Stable
    - Usable
  - Taxonomy_Mapping
  - View
    - Explicit
    - Graph
    - Implicit
  - Weakness

- owl:topObjectProperty
  - CAPEC_ID
  - category
  - Consequence
  - Execution_Flow
  - Has_Member
  - Member_Of
  - Note
  - Related_Attack_Pattern
    - CanAlsoBe
    - CanFollow
    - CanPrecede
    - ChildOf
    - ParentOf
    - PeerOf
  - Related_Weakness
  - Taxonomy_Mapping
  - Technique

- owl:topDataProperty
  - Alternate_Term
  - Audience
  - Consequence_ID
  - Entry_ID
  - Entry_Name
  - ID
  - Impact
  - Level
  - Likelihood
  - Likelihood_Of_Attack
  - Mapping_Fit
  - Name
  - Phase
  - Scope
  - Skill
  - Step
  - Taxonomy_Name
  - Type
  - Typical_Severity

Figure 7. CAPEC classes, object and data properties

The NVD ontology is divided in separate ontologies – one by every year and one for Modified CVEs. The name of this ontology is CVE.

The ontology of the year is subdivided in parts to be manageable in Protégé. Every part has a set of separate ontologies containing references to vulnerable CPE configurations.

All parts can be opened in Protégé separately including the parts in the above presented order.

The definitions (classes, object and data properties) of this ontology are given in the NVD ontology. They are represented in Figure 3.

Class CVE refers to its vulnerable configurations via object property "configuration" to individuals of the class Configuration.

Class CVSSV20 and CVSS3X are used to present CVE scoring via object property referencing.

Classes Product, Vendor and Version are used for backward compatibility.

Figure 4 illustrate a CVE individual.

Detailed presentation of the CVE ontology and its generator will be published soon.

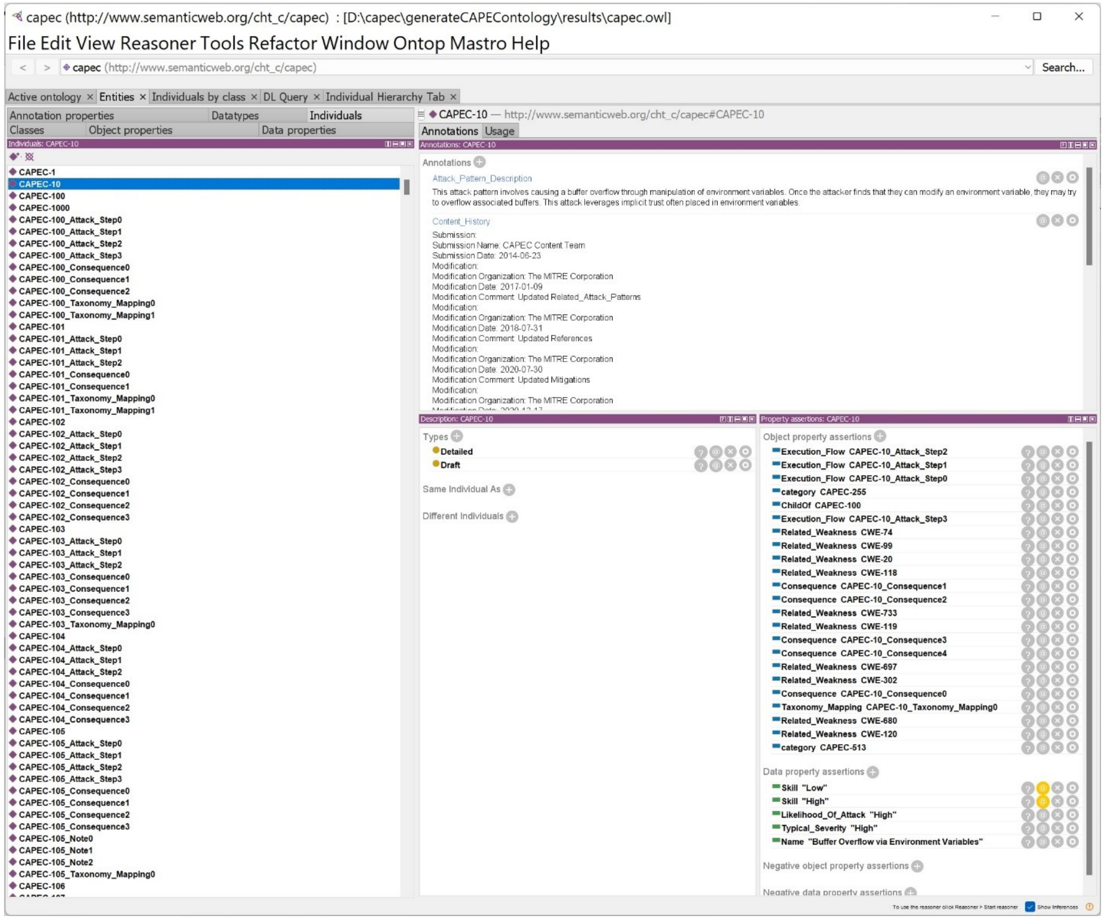
2.3. CWE ontology

The CWE ontology is more complex but fortunately not as huge as the CVE one. Here only a brief description of this ontology is presented.

Views are disjoint union of their subclasses.

Class Category is simply a container as in the CWE database.

Weakness is disjoint union of its subclasses, but it is equivalent to Structure. The last one is a disjoint union of its subclasses.



capec (http://www.semanticweb.org/cht_c/capec) : [D:\capec\generateCAPEContology\results\capec.owl]

File Edit View Reasoner Tools Refactor Window Ontop Mastro Help

capec (http://www.semanticweb.org/cht_c/capec)

Active ontology × Entities × Individuals by class × DL Query × Individual Hierarchy Tab ×

Annotation properties Datatypes Individuals

Classes Object properties Data properties

Individuals: CAPEC-10

CAPEC-1
CAPEC-10
CAPEC-100
CAPEC-1000
CAPEC-100_Attack_Step0
CAPEC-100_Attack_Step1
CAPEC-100_Attack_Step2
CAPEC-100_Attack_Step3
CAPEC-100_Consequence0
CAPEC-100_Consequence1
CAPEC-100_Consequence2
CAPEC-100_Taxonomy_Mapping0
CAPEC-100_Taxonomy_Mapping1
CAPEC-101
CAPEC-101_Attack_Step0
CAPEC-101_Attack_Step1
CAPEC-101_Attack_Step2
CAPEC-101_Consequence0
CAPEC-101_Consequence1
CAPEC-101_Taxonomy_Mapping0
CAPEC-101_Taxonomy_Mapping1
CAPEC-102
CAPEC-102_Attack_Step0
CAPEC-102_Attack_Step1
CAPEC-102_Attack_Step2
CAPEC-102_Attack_Step3
CAPEC-102_Consequence0
CAPEC-102_Consequence1
CAPEC-102_Consequence2
CAPEC-102_Consequence3
CAPEC-103
CAPEC-103_Attack_Step0
CAPEC-103_Attack_Step1
CAPEC-103_Attack_Step2
CAPEC-103_Consequence0
CAPEC-103_Consequence1
CAPEC-103_Consequence2
CAPEC-103_Consequence3
CAPEC-103_Taxonomy_Mapping0
CAPEC-104
CAPEC-104_Attack_Step0
CAPEC-104_Attack_Step1
CAPEC-104_Attack_Step2
CAPEC-104_Consequence0
CAPEC-104_Consequence1
CAPEC-104_Consequence2
CAPEC-104_Consequence3
CAPEC-105
CAPEC-105_Attack_Step0
CAPEC-105_Attack_Step1
CAPEC-105_Attack_Step2
CAPEC-105_Attack_Step3
CAPEC-105_Consequence0
CAPEC-105_Consequence1
CAPEC-105_Consequence2
CAPEC-105_Consequence3
CAPEC-105_Note0
CAPEC-105_Note1
CAPEC-105_Note2
CAPEC-105_Taxonomy_Mapping0
CAPEC-106

CAPEC-10 — http://www.semanticweb.org/cht_c/capec#CAPEC-10

Annotations Usage

Annotations: CAPEC-10

Annotations

Attack_Pattern_Description

This attack pattern involves causing a buffer overflow through manipulation of environment variables. Once the attacker finds that they can modify an environment variable, they may try to overflow associated buffers. This attack leverages implicit trust often placed in environment variables.

Content_History

Submission:
Submission Name: CAPEC Content Team
Submission Date: 2014-06-23
Modification:
Modification Organization: The MITRE Corporation
Modification Date: 2017-01-09
Modification Comment: Updated Related_Attack_Patterns
Modification:
Modification Organization: The MITRE Corporation
Modification Date: 2018-07-31
Modification Comment: Updated References
Modification:
Modification Organization: The MITRE Corporation
Modification Date: 2020-07-30
Modification Comment: Updated Mitigations
Modification:
Modification Organization: The MITRE Corporation

Description: CAPEC-10

Types
Detailed
Draft

Same Individual As

Different Individuals

Property assertions: CAPEC-10

Object property assertions
Execution_Flow CAPEC-10_Attack_Step2
Execution_Flow CAPEC-10_Attack_Step1
Execution_Flow CAPEC-10_Attack_Step0
category CAPEC-255
ChildOf CAPEC-100
Execution_Flow CAPEC-10_Attack_Step3
Related_Weakness CWE-74
Related_Weakness CWE-99
Related_Weakness CWE-20
Related_Weakness CWE-118
Consequence CAPEC-10_Consequence1
Consequence CAPEC-10_Consequence2
Related_Weakness CWE-733
Related_Weakness CWE-119
Consequence CAPEC-10_Consequence3
Consequence CAPEC-10_Consequence4
Related_Weakness CWE-697
Related_Weakness CWE-302
Consequence CAPEC-10_Consequence0
Taxonomy_Mapping CAPEC-10_Taxonomy_Mapping0
Related_Weakness CWE-680
Related_Weakness CWE-120
category CAPEC-513

Data property assertions
Skill "Low"
Skill "High"
Likelihood_Of_Attack "High"
Typical_Severity "High"
Name "Buffer Overflow via Environment Variables"

Negative object property assertions

Negative data property assertions

To use the reasoner click Reasoner > Start reasoner ✓ Show Inferences

Figure 8. CAPEC individual

Status is a disjoint union of its subclasses, but also a disjoint union of Category, View and Weakness.

`Applicable_Platform`, `Consequence`, `Demonstrative_Example`, `Note`, `Detection_Method`, `Observed_Example`, `Potential_Mitigation`, and `Taxonomy_Mapping` are classes describing some aspects of CWE with individuals.

`Has_Member`, `Member_Of`, `Related_Weakness` (more specifically its subproperties) are object properties that have meaning only in the context of a given view. That is why for every view there are generated subproperties with the same name but in a name space corresponding to the view.

Definitions of CWE classes, object and data properties and some individuals are presented in Figures 5 and 6.

2.4. CAPEC ontology

The CAPEC ontology looks pretty much as the CWE ontology, but it is smaller and simpler.

Here, the ontology is a collection of views that can be structured in categories. Instead of a Weakness class an `Attack_Pattern` is used and its subclasses are Meta, Standard, and Detailed.

Figure 7 and Figure 8 represent definitions and individuals of this ontology.

More details on the CAPEC ontology and its generator will be published soon.

3. Conclusion

Currently the main problem is how to load all these ontologies into one triple store that has reasoners. The usage of the last ones is the main motivation for this research.

Most of the triple stores support only SPARQL as a query language but no reasoners.

Another problem is that the OWL Manchester syntax is not supported by the triple stores. This means that these ontologies must be converted to some other syntax before they can be downloaded. Tools for automatic conversation exist, but their usage hardly can be automated.

Finally, NVD, CPE, CWE, and CAPEC are under permanent improvement and development. This means that the ontologies and the generators for them must follow source changes.

References

[1] R. L. Ackoff, From data to wisdom, Journal of Applied Systems Analysis 16 (1989) 3–9.

[2] V. Dimitrov, CPE Ontology, in: Proc. of ISGT'2021, Sofia, Bulgaria, May 28–29, 2021, 302–313, `https://ceur-ws.org/Vol-2933/paper30.pdf`.

[3] V. Dimitrov, CPE Ontology Generator, in: Proc. of ISGT'2021, Sofia, Bulgaria, May 28–29, 2021, 359–366, `https://ceur-ws.org/Vol-2933/paper34.pdf`.

[4] FIRST, Common Vulnerability Scoring System SIG, `https://www.first.org/cvss`.
[5] K. C. Laudon and J. P. Laudon, Management Information Systems. Managing the Digital Firm, Fourteenth Edition, Global Edition, Pearson Education Limited, 2016.
[6] The MITRE Corporation, CVE (Common Vulnerabilities and Exposures), `https://cve.mitre.org`.
[7] The MITRE Corporation, CVE (Common Vulnerabilities and Exposures), `https://www.cve.org`.
[8] The MITRE Corporation, CWE (Common Weakness Enumeration), `https://cwe.mitre.org`.
[9] The MITRE Corporation, CAPEC (Common Attack Pattern Enumeration and Classification), `https://capec.mitre.org`.
[10] M. A. Musen, The protégé project: a look back and a look forward, AI Matters 1(4) (2015) 4–12, `https://dl.acm.org/doi/10.1145/2757001.2757003`.
[11] NIST Information Technology Laboratory, NVD, `https://nvd.nist.gov`.
[12] NIST Information Technology Laboratory, NVD, Official Common Platform Enumeration (CPE) Dictionary, `https://nvd.nist.gov/products/cpe`.
[13] W3C, Semantic Web, `https://www.w3.org/standards/semanticweb`.
[14] W3C, OWL 2 Web Ontology Language, Manchester Syntax (Second Edition), W3C Working Group Note 11 December 2012, `https://www.w3.org/TR/owl2-manchester-syntax/#:~:text=TheManchestersyntaxisa,providesthelanguageusedthere`.

Received on November 20, 2022
Revised on December 6, 2022
Accepted on December 8, 2022

Vladimir Dimitrov
Faculty of Mathematics and Informatics
Sofia University "St. Kliment Ohridski"
5 James Bourchier Blvd.
1164 Sofia
BULGARIA
E-mail: `cht@fmi.uni-sofia.bg`