

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



SÉMANTICKÉ PUBLIKOVANIE SPRAVODAJSKÝCH DÁT O BEZPEČNOSTNÝCH HROZBÁCH

Diplomová práca

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



SÉMANTICKÉ PUBLIKOVANIE SPRAVODAJSKÝCH DÁT O BEZPEČNOSTNÝCH HROZBÁCH

Diplomová práca

Študijný program: Aplikovaná informatika
Študijný odbor: 2511 Aplikovaná informatika
Školiace pracovisko: Katedra aplikovanej informatiky
Školiteľ: doc. RNDr. Martin Homola, PhD.
Konzultant: Ing. Štefan Balogh, PhD.

Bratislava, 2021

Bc. Matej Rychtárik



Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Matej Rychtárik
Študijný program: aplikovaná informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Sémantické publikovanie spravodajských dát o bezpečnostných hrozbách
Semantic punishing of security threat intelligence data

Anotácia: V súčasnosti zaznamenávame veľké množstvo nových spravodajských dát o rôznych bezpečnostných hrozbách. Pre popis a publikovanie týchto dát vznikli v minulosti viaceré štandardy. Nový trend v oblasti však ukazuje potrebu sémantickej anotácie týchto dát za účelom zvýšenia ich dosahu a interoperability.

Cieľ: Cieľom je navrhnúť vhodnú ontológiu pre publikovanie spravodajských dát o bezpečnostných hrozbách a vytvorenie repozitára za týmto účelom v sieti prepojených dát.

Literatúra: [1] Allemang, D. and Hendler, J., 2011. Semantic web for the working ontologist: effective modeling in RDFS and OWL. Elsevier.
[2] Heath, T. and Bizer, C., 2011. Linked data: Evolving the web into a global data space. Synthesis lectures on the semantic web: theory and technology, 1(1), pp.1-136.
[3] Mavroeidis, V. and Bromander, S., 2017. Cyber threat intelligence model: an evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence. In EISIC 2017 (pp. 91-98). IEEE.

Vedúci: doc. RNDr. Martin Homola, PhD.
Rektorát, dekanát: FMFI.Dek - Dekanát
Dátum zadania: 02.10.2019

Dátum schválenia: 14.10.2019
prof. RNDr. Roman Ďurikovič, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Čestne prehlasujem, že túto diplomovú prácu som
vypracoval samostatne len s použitím uvedenej literatúry
a za pomoci konzultácií u môjho školiteľa.

Bratislava, 2021

.....

Bc. Matej Rychtárik

Pod'akovanie

Touto cestou by som sa chcel v prvom rade poďakovať môjmu školiteľovi doc. RNDr. Martinovi Homolovi, PhD. za jeho cenné rady a usmernenia, ktoré mi veľmi pomohli pri riešení tejto diplomovej práce.

MH: Nezabudni podakovat sa aj Stefanovi. Inak tu tituly mozes vypustit, my dvaja si ozaj na to nepotrpime ;)

Abstrakt

Abstrakt v slovenčine

Abstract

An abstract in english language

Obsah

1	Úvod	1
I	Prehľad problematiky	2
2	Sémantický web	3
2.1	Linked Data	5
2.2	Resource Description Framework (RDF)	6
2.3	SPARQL	9
3	Ontológie	11
3.1	Základné pojmy	12
3.2	Využitie ontológií	13
3.3	Web Ontology Language	15
3.3.1	Syntax	16
4	Existujúce ontologické riešenia	21
4.1	CTI model	22
4.1.1	Identita	23
4.1.2	Kampaň	24
4.1.3	Zraniteľnosť	24
4.1.4	Indikátor	25

<i>OBSAH</i>	ix
--------------	----

4.1.5	Nástroj	25
4.1.6	Zámer	26
4.1.7	Cieľ	26
4.2	Unified Cybersecurity Ontology	26
4.3	Integrated Cyber Analysis System	28
4.4	STUCCO	28

II Vlastný prínos 30

5 Výskum a Analýza UCO 31

5.1	Model	32
-----	-----------------	----

6 Návrh ontológií 39

6.1	CVE	39
6.2	OVAL	41

7 Implementácia a testovanie 45

7.1	Dáta	45
7.2	Testovanie	46
7.2.1	Príklady použitia	46
7.3	Internetová stránka	52

8 Záver 53

Kapitola 1

Úvod

Nejaky strucny uvod do problematiky

How to write an introduction – some suggestions:

<https://www.win.tue.nl/~setalle/introduction.html>

Časť I

Prehľad problematiky

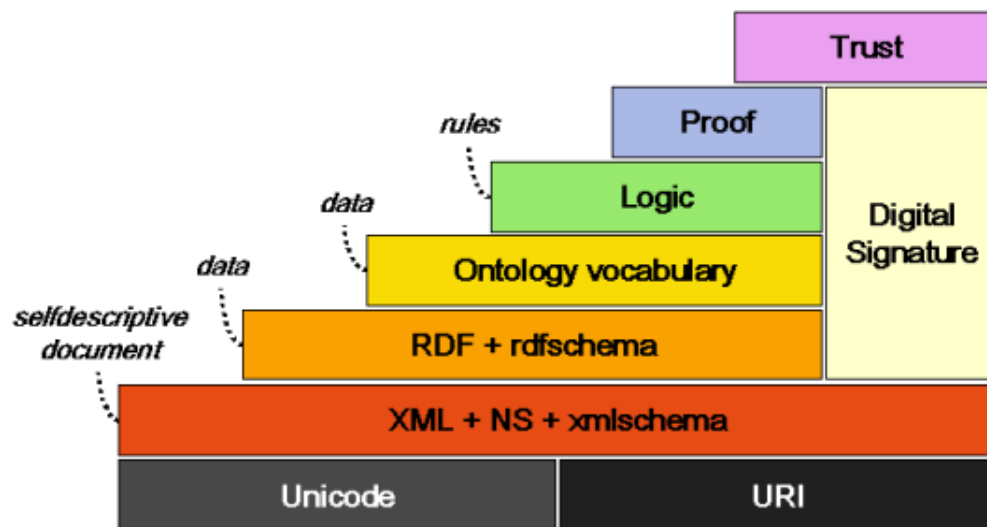
Kapitola 2

Sémantický web

Sémantický web [BLHL01] poskytuje spoločný framework, ktorý umožňuje zdieľanie a opätovné použitie údajov v rámci aplikácií. Štandardy podporujú spoločné dátové formáty a protokoly, kde najpodstatnejším je Resource Description Framework (RDF). Prvýkrát pojem Sémantický web zaviedol Tim Berners-Lee a popisoval "dátový web", ktorý môže byť strojovo čitateľný. Zámerom je zlepšiť prístupnosť informácií publikovaných na webe pre strojové spracovanie. Sémantický web má vrstvovú štruktúru ako si môžeme všimnúť na obrázku 2.1. Jednotlivé údaje sú potrebné až vo vyšších vrstvách.

XML vrstva zaručuje, že môžeme spájať Sémantický web s inými normami, založenými napríklad na XML, ktorá je rozšírená a podporovaná a RDF dáta sa v nej dajú dobre prenášať, spracovávať a uchovávať. Toto je už ale pre dnešné časy neštandardné a viac sa využíva výmena dát pomocou JSON formátu.

RDF je metóda popisovania vecí pomocou vzťahu medzi dvoma objektami. Napríklad koncepčne povedané "Jožko má jablko" je zadané definovaním spojenia medzi objektami "Jožko" a "jablko" pomocou



Obr. 2.1: Semantic Web - vrstvy.
Zdroj: [KM]

vzťahu "má". Toto spojenie je známe ako triplet, ktorý je základnou stavebnou jednotkou sémantického webu.

RDF je aj názov slovníka, ktorý obsahuje množinu preddefinovaných termínov. Tieto termíny sú všeobecne používané na popis dát. Napríklad obsahuje najzákladnejšiu vlastnosť pre objekty a to vlastnosť typu objektu – *rdf:type*.

RDFS je taktiež slovník. V RDF slovníku máme termíny, ktoré nám pomáhajú určovať definície a popisy jednotlivých objektov. V RDFS získavame možnosť popisovať triedy. Pokiaľ si zoberieme triedu "Osoba" a triedu "Žena", vieme pomocou RDFS slovníka zadefinovať vzťah podtriedy vlastnosťou "rdfs:subClassOf". Vďaka slovníkom RDF a RDFS môžeme tvoriť detailné popísanie našich dát.

Ontológia je v našom ponímaní synonymom k slovu slovník. Slovník RDFS môže byť použitý na tvorbu vlastnej ontológie. Sú v nej definované

Vďaka takejto reprezentácii dát je možné písať pravidlá, ktoré má daný

súbor dát spĺňať. Vďaka logickej vrstve vieme napríklad zdefinovať pravidlá ako: Všetky objekty typu "Muž" a "Žena" sú zároveň typom "Osoba" alebo Množina objektov s typom "Muž" je disjunktná s množinou objektov "Žena". Tieto pravidlá slúžia na kontrolu konzistentnosti našich dát.

MH: ↑Toto trochu povrchné: (1) Ucelom SW nie je vyššia použiteľnosť webu (to je nepresne), ale je to lepšia prístupnosť informácií publikovaných na webe pre strojové spracovanie. (2) Ak chceš popisovať vrstvy SW podľa tohto diagramu, bolo by dobre keby si popísal všetky vrstvy – U XML by som sa obmedzil na to, že je to prostý dobrý formát pre textovú reprezentáciu dát v súboroch a pre ich výmenu medzi softvermi – toto však už je dnes prekonané, už vymeníme SW dáta aj ako JSON, embedujeme ich do HTML5 (chcelo by to poznámku) – O RDF a RDFS si vlastne nič užitočné (z čoho čitateľ niečo vyrozumie) nepovedal – no a ostatné vrstvy si úplne preskocil

MH: Toto si menil? Zdá sa mi, že trochu asi aj áno, ale o tých ďalších vrstvách si nič nenapísal

MR: Popísané sú už aj ďalšie vrstvy, mali by byť už hotové

Text uvedený nižšie popisuje niekoľko technológií, ktoré sú potrebné pre tvorbu sémantického webu.

2.1 Linked Data

MH: ↓Tato sekcia je celkom fajn ale chýbalo mi trochu premostenie od SW – linked data bola iniciatíva, že keď už SW formáty máme, podme v nich aj dáta zverejňovať

Linked Data [BHBL11] alebo prepojené dáta, je metóda zverejňovania štrukturovaných dát. Ich hlavným cieľom je poprepájať existujúce databázy (primárne písané v RDF formáte), medzi rôznymi údajmi a umožniť ľuďom zdieľať štrukturované dáta na webe pomocou HTML. Časť vízie do budúcnosti je, aby sa Internet stal globálnou databázou. Princípy prepojených dát prvýkrát načrtol Tim Berners-Lee. Popísal 4 pravidlá pre zverejňovanie dát na webe:

1. používať URI ako názvy objektov, ktoré sú identifikátormi informácie, jej umiestnenia a ďalších vlastností,
2. používať HTTP URI, aby si ich ľudia vedeli pozrieť,
3. uvádzať informácie o tom, čo názov identifikuje pri vyhľadávaní pomocou otvorených štandardov, ako sú napríklad RDF alebo SPARQL,
4. pri publikovaní údajov na webe, zahrnúť odkazy aj na iné URI, aby sa dalo objavovať viac vecí.

Sú známe aj ako Princípy prepojených dát.

MH: Tu mi chyba informacia, že sa táto iniciatíva ujala, a že vďaka tomu vznikla na webe tzv. sieť prepojených dát, ktorá obsahuje obrovské množstvo dátových zdrojov a niečo viac o tej sieti.

2.2 Resource Description Framework (RDF)

RDF [SRb] je štandardný model na zakódovanie metadát a ďalších informácií. Je to taktiež formát, ktorý bol navrhnutý a štandardizovaný na reprezentáciu dát pre sémantický web. Zdroje týchto dát sú väčšinou webové zdroje, ktoré môžu byť čokoľvek, napríklad dokumenty, ľudia, fyzické objekty, atď. Taktiež poskytuje spoločný framework na vyjadrenie informácií a možnosť zdieľať ich medzi softvérmi, bez straty ich hodnoty. Dáta sa uchovávajú v Triple Store databázach, ktorých formát je striktne daný. Výhodou je, že dáta môžu byť spracované aj softvérmi, pre ktoré dané dáta neboli vytvorené.

MH: ↑Na co RDF sluzi sa uz citatel dozvedel v skorsich castiach (ked to tam lepsie ozrejmis). Niektore veci, ktore tu ↑pises su nepresne (napr. to o tych metadatach a “dalsich informaciach” alebo o zdrojoch. Tiez o Triple Stores predbiehas, budes o tom pisat neskor. . . Asi by som to tu skratil a len by som nadviazal, ze RDF je zakladny datovy format pre SW a tu ho popiseme. . .

MH: ↓Zvysok ide dobrym smerom, je to presne to, co by som si predstavoval, ze tu budes pisat, len by som to chcel vidiet mozno trochu pomenej, podrobnejsie, systmatickejsie prebrate. . . Na vacsom priestore, mozno postupne ten priklad budovat. . . Vysvetlit na nom vsetky zakladne moznosti RDF

RDF súbor je taký dokument, ktorý ukladá RDF grafy do špecifického formátu serializácie pre RDF, ako sú napríklad N-Triple, Turtle, RDF/XML a mnohé ďalšie. RDF bol postavený na myšlienke vytvárať údaje vo forme predmet-predikát-objekt, ktorý sa volá triplet. Triplet je základná stavebná jednotka akejkolvek množiny dát zapísaných v RDF. Tieto údaje sú reprezentované ako orientované grafy. Predmet a objekt predstavujú vrcholy a predikát je orientovaná hrana medzi nimi. Predmet môže byť použitý aj ako objekt v inom triplete. Týmto spôsobom sa triplety prepájajú a vzniká z nich grafová databáza. Predmet je vždy definovaný ako URI a popisuje zdroj informácie. Objekt môže byť taktiež nejaké URI popisujúce zdroj, ale taktiež to môže byť primitívna hodnota, ako napríklad string, integer, date, atď. Predikát popisuje, aký vzťah alebo rola medzi predmetom a objektom existuje. Predikát je vždy reprezentovaný ako URI, ktoré pochádza z ontológií (kolekcie viacerých URI).

Na uľahčenie ukladania a čitateľnosti dát sa využívajú takzvané prefixy, ktoré sú preddefinovaním základných URI, do ktorých sa dodáva zvyšná hodnota URI pomocou dvojbodky, ako je to uvedené v nasledujúcom príklade a graficky znázornené v obrázku 2.2.


```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns> .
```

```
@prefix dbr: <http://dbpedia.org/resource/> .
```

```
@prefix dbo: <http://dbpedia.org/ontology/> .
```

```
@prefix dbp: <http://dbpedia.org/property/> .
```

```
dbr:Bratislava dbo:highestPlace dbr:Devínska_Kobyla .
```

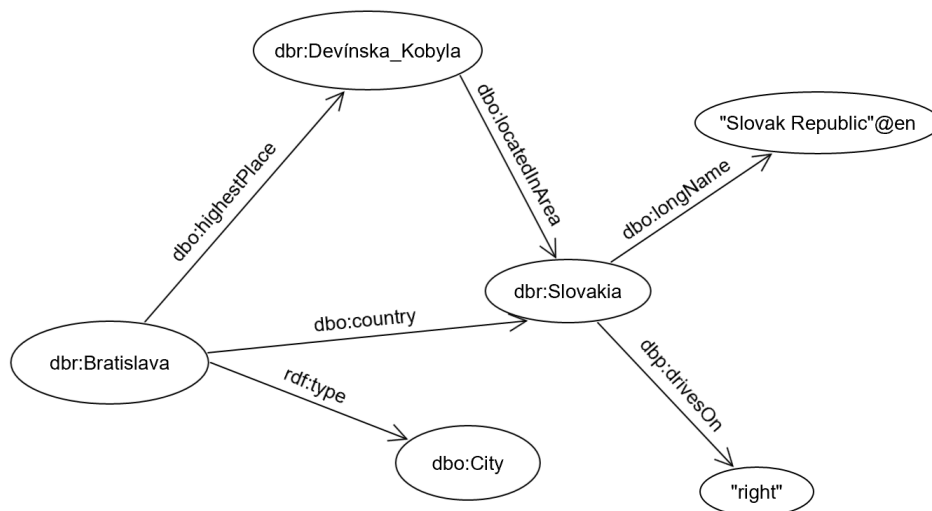
```
dbr:Bratislava rdf:type dbo:City .
```

```
dbr:Bratislava dbo:country dbr:Slovakia .
```

```
dbr:Devínska_Kobyla dbo:locatedInArea dbr:Slovakia .
```

```
dbr:Slovakia dbp:drivesOn "right" .
```

```
dbr:Slovakia dbo:longName "Slovak Republic"@en .
```



Obr. 2.2: Príklad grafovej databázy.

2.3 SPARQL

SPARQL [SRa] je dopytovací jazyk pre RDF databázy, ktorý umožňuje získavanie a manipuláciu s databázou. Bol vytvorený skupinou DAWG, ktorá je súčasťou W3C a je uznávaný ako kľúčová technológia sémantického webu.

Ak by sme porovnali SPARQL s dopytovacím jazykom pre relačné databázy, napr. SQL, zistíme, že sú si podobné v kľúčových slovách, ako sú napr. SELECT, WHERE, FROM atď. SPARQL dopyt využíva trojice ako základný prvok, kde predmet, predikát alebo objekt môžu byť premenné. Dopyt sa robí nad dátovou kolekciou RDF, čo je množina dokumentov, patriaca pod určitý koncový bod - '*endpoint*'. Je to dopytovací jazyk, ktorý z orientovaného ohodnoteného grafu zisťuje hodnoty jednotlivých vrcholov a hrán, ktoré sú výstupnými parametrami dopytu.

```
@prefix dbr: <http://dbpedia.org/resource/> .
```

```
SELECT ?predicate ?object WHERE {
    dbr:Bratislava ?predicate ?object .
}
```

```
+-----+-----+
| ?predicate      | ?object          |
+-----+-----+
| dbo:highestPlace | dbr:Devínska_Kobyla |
| rdf:type         | dbo:City          |
| dbo:country      | dbr:Slovakia      |
+-----+-----+
```

Príklad dopytu nad databázou uvedenou vyššie, spúšťame nad endpointom DBPedia a výsledok je len zlomkom z toho, čo nám skutočne

vráti: Chceme získať všetky údaje o Bratislave.

Okrem operácie SELECT poznáme aj ďalšie typy dopytov. ASK je dopyt, ktorý nám vracia pravdivostnú hodnotu pre daný dopyt. Vieme ním napríklad zistiť či sa v našom grafe nachádza mesto Bratislava. Taktiež poznáme dopyt DESCRIBE, ktorý vracia RDF graf opisujúci jednotlivé vlastnosti výsledných hodnôt dopytu. Ako posledný typ dopytu je CONSTRUCT, ktorý vracia nový RDF graf podľa predlohy vytvorenej v hlave dopytu.

Kapitola 3

Ontológia

MH: Predpokladam, že o ontológiach budeme písať o značný kus viac a podrobnejšie. . . Možno vychádzať z mojej prednášky ale tiež napríklad z úvodnej kapitoly *Handbook on Ontologies*

Výraz ontológia [GOS09] pochádza z gréckeho slova kde 'ontos' znamená existencia a 'logos' znamená veda. Ontológia v informatike je uceleným popisom pojmov v určitej oblasti záujmu. Obsahuje určitú klasifikáciu údajov do hierarchicky usporiadaných kategórií a množinu odvodzovacích pravidiel, pomocou ktorých je možné z faktov odvodiť nové skutočnosti. Prostredníctvom ontológií je možné vytvárať spojenia, vykonávať analýzu údajov a sprostredkovať výhody webu obohateného o sémantiku.

Jej cieľmi je zadefinovanie a zdieľanie jednotného zápisu informácií pre danú doménu. Ak napríklad viac stránok využíva na popis pojmov takúto zadefinovanú ontológiu, vedia ju získať a vyhľadávať viac dát o hľadanej informácii.

Taktiež je jej cieľom opätovné použitie ontológie, napríklad ak máme dobre zadefinovanú ontológiu, môžu ontologickí inžinieri doplniť do našej

ontológie ďalšie vlastnosti a tým by základ ontológie bol rovnaký ale bol by rozšírený o určité dáta, podľa potreby ontologických inžinierov.

3.1 Základné pojmy

Ontológia sa skladá zo základných stavebných prvkov *Trieda*, *Entita*, *Atribút*, *Vzťah*.

Triedy alebo typy definujú skupiny alebo množiny objektov. Triedy majú hierarchickú štruktúru zloženú z ich podtried. Každá podtrieba spĺňa vlastnosti nadtriedy a môže byť rozšírená o vlastné vlastnosti.

Entity sú individuálne inštancie nejakej nami zadanovej triedy. Ak by sme mali entitu *Bratislava*, a triedy *Mesto* a *Hlavné mesto*, kde *Mesto* je podtriedou *Hlavné mesto*, tak nám z ontológie vyplýva, že ak je entita *Bratislava* individuálnou inštanciou triedy *Hlavné mesto*, tak je aj individuálnou inštanciou triedy *Mesto*.

Atribúty sú vlastnosti *Tried* a *Entít* a môžu niesť rôzne informácie o danom objekte. *Atribúty* môžu mať rôzne hodnoty, ako reťazec, číslo, dátum alebo pravdivostnú hodnotu. Ak by sme si zobrali predchádzajúcu entitu *Bratislava*, jej číselná vlastnosť môže byť napríklad počet obyvateľov.

Vzťahy sú najpodstatnejšou súčasťou ontológie. Poskytujú prepájanie jednotlivých entít. Je to jednosmerné spojenie, ktoré určuje vzťah, v akom sú dve dané triedy. Tým vznikne triplet *trieda:vzťah:trieda*. Medzi triplet sa radí aj trojica *trieda:atribút:hodnota*. Väzby sa zvyknú definovať aj inverzne. Z logického hľadiska sú vzťahy axiómami. Pokiaľ máme triedu *Krajina* a *Hlavné mesto*, tak by vzťah mohol vyzeráť nasledovne: *Krajina:má:Hlavné mesto*.

TBox je množina definícií tried a ich vzťahov medzi nimi. V množine

je obsiahnutá znalosť taxonómie tried, taktiež v nej môže byť zadaná disjunktnosť jednotlivých tried, vymenovanie konkrétnych entít obsiahnutých v danej triede, reštrikcie pre jednotlivé triedy a ich vzťahy

ABox je množina znalostí o jednotlivých entitách.

Ontológia má veľa vlastností, ktoré musia byť dodržané. Každý prvok musí byť jasne identifikovateľný. Taktiež zakazuje zapisovanie duplicitných dát, čo nám zaobstará vlastnosť efektívneho ukladania informácií, kde to môže nie len uľahčiť vyhľadávanie ale aj zredukovať obsah pamäti na disku.

MH: ↑ Neviem, či zrovna tieto vlastnosti ontológii sú tie najpodstatnejšie, a teda treba ich spomínať na tomto mieste.

MR: Určite by som ich spomenul nakoľko existuje určite veľa duplicitných dát a je to výhoda oproti SQL kde každý záznam v číselníku má unikátne IDčko ale u nás je to predstavované iba entitou.

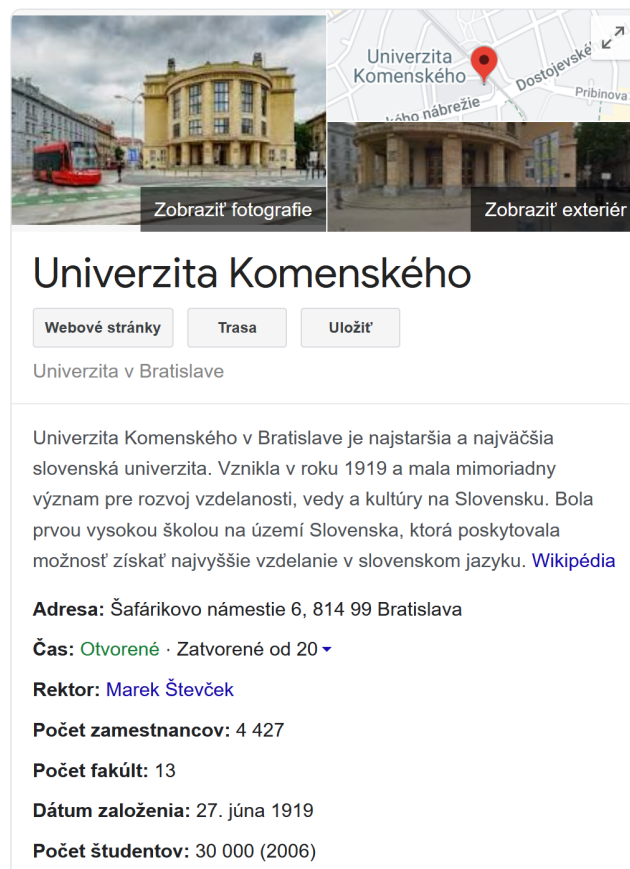
3.2 Využitie ontológií

Ontológie sa začali využívať najmä v organizáciách, ktoré sa špecializovali na umelú inteligenciu. Neskôr sa to rozšírilo aj do bežne používaných aplikácií. Napríklad firma Amazon používa ontológie na kategorizovanie tovaru v ich elektronickom obchode.

MH: ↑ Vies to doložiť nejakou referenciou? Skade máš tú informáciu...

Ontológie si našli uplatnenie aj v medicínskej oblasti a to napríklad SNOMED, čo je najväčším viacjazyčným medicínskym slovníkom na svete.

Taktiež sa s ním stretávame každodenne pri vyhľadávaní na stránke Google, kde ako bočný panel sú zobrazené informácie o vyhľadávanom objekte (??). Tieto dáta je možné zobrazovať preto, lebo výsledkom takéhoto panelu je vyhľadávanie informácií na webovej stránke, ktorá obsahuje sémantické dáta.



Obr. 3.1: Príklad bočného panelu vo vyhľadávачi Google.

Schema.org [sch] je taktiež ontologickým riešením, ktorej cieľom je vytvárať, udržiavať a propagovať schémy štrukturovaných údajov na internete, internetových stránkach alebo e-mailových správach. Poskytuje zdieľanú slovnú zásobu, ktorú môžu tvorcovia internetových stránok používať na označenie a popis jednotlivých elementov na stránke. Týmto popisom rozumejú vyhľadávacie nástroje ako Google, Microsoft a Yahoo!. Viac ako 10 miliónov internetových stránok používa Schema.org na popisovanie svojich stránok a e-mailových správ.

MH: ↑ SNOMED a Google spomínas dobre, a v druhom prípade si nespomenul žiadnu ontológiu a pritom ju dobre pozname (Schema.org)

MR: spomenute

MH: ↑ toto mi sem opat nesedi, zrejme tomu chces venovať nejakú kratku podsekcii skor v predchádzajúcej kapitole (?)

MR: ja som ju tam mal len predtým si mi písal že si si este nie istý či nás to bude zaujímať, tak som to vrátil do predchádzajúcej časti

MH (21.4.): OK, len stále tu píšeš základné informácie o Syntaxi SPARQL, to tu zrejme nemá byť

MR: okej upravil som

MH (26.4.): Asi stále nerozumiem ako tá posledná časť o datach a SPARQL súvisí s touto sekciou... Proste nejak zrazu prepneš na data a SPARQL od konkrétnych ontológií, ktoré si vmenuješ, chyba premostenie

MR: jop, už chápem, odstránil som tú časť

3.3 Web Ontology Language

Jazyk Web Ontology Language alebo OWL slúži na vytváranie a definovanie inštancií webových ontológií. Poskytuje nástroje na popis tried, vlastností a ich samotných inštancií. Oproti klasickým jazykom poskytuje možnosť, špecifikovať logické vlastnosti jednotlivých objektov vyskytujúcich sa v ontológií, to znamená, že dokáže popísať aj fakty, ktoré v ontológií nie sú definované priamo, ale sú spojené logickými vlastnosťami. Napríklad, ak máme osoby Janka, Martin a Jozef, kde Janka je rodičom Martina a Jozef je rodičom Janky, pomocou OWL vieme definovať že Jozef je starým rodičom Martina, pričom nepotrebujeme vlastnosť popisujúcu tento vzťah priamo.

MH: ↑ Tomuto nerozumiem

3.3.1 Syntax

MH: (1) Man syntax je zrozumiteľnejšia (2) moc daleko si nedosiel... Na zač. sa detailne venujeme nie príliš podstatným veciam

Na popis jednotlivých možností budeme používať Manchester syntax, ktorá poskytuje zrozumiteľnejšiu syntax pre čitateľa.

Ako prvé si zadefinujeme priradenia objektov ku jednotlivým triedam. Objekty môžu mať pridelených viac tried.

```
ClassAssertion( :Osoba :Janka )
ClassAssertion( :Žena :Janka )
ClassAssertion( :Osoba :Martin )
ClassAssertion( :Muž :Martin )
ClassAssertion( :Osoba :Jozef )
ClassAssertion( :Muž :Jozef )
```

Ako si môžeme všimnúť vyššie, Janke sme museli priradiť dve triedy. Týmto triedam ale vieme vytvoriť hierarchickú štruktúru.

```
SubClassOf( :Žena :Osoba )
SubClassOf( :Matka :Žena )
```

Niekedy môžeme mať dve rôzne definície tried, ktoré ale predstavujú tú istú osobu. Takéto triedy sa volajú ekvivaletné a vieme ich definovať nasledovne:

```
EquivalentClasses( :Človek :Osoba )
```

Ďalej si vieme zadefinovať vlastnosti objektov. Tieto vlastnosti vieme definovať buď pozitívne, čiže objekt danú vlastnosť má alebo negatívne, objekt danú vlastnosť s druhým objektom mať nemôže.

```
ObjectPropertyAssertion( :jeMatkou :Janka :Martin )
```

```
ObjectPropertyAssertion( :jeOtcom :Jozef :Janka )
```

```
NegativeObjectPropertyAssertion( :jeOtcom :Jozef :Martin )
```

Veľmi podobne vieme definovať aj dátové priradenia k jednotlivým objektom.

```
DataPropertyAssertion( :máVek :Janka "38"^^xsd:integer )
```

```
NegativeDataPropertyAssertion( :máVek :Jozef "10"^^xsd:integer )
```

Tieto vlastnosti môžu mať taktiež hierarchickú štruktúru.

```
SubObjectPropertyOf( :jeMatkou :jeRodičom )
```

```
SubObjectPropertyOf( :jeOtcom :jeRodičom )
```

Týmto vlastnostiam vieme definovať ich domény a rozsah, ktoré definujú aké hodnoty môžu jednotlivé vlastnosti nadobúdať a z akým objektom môžu byť tieto vlastnosti pridelené.

```
ObjectPropertyDomain( :jeRodičom :Osoba )
```

```
ObjectPropertyRange( :jeRodičom :Osoba )
```

```
ObjectPropertyDomain( :jeMatkou :Žena )
```

```
ObjectPropertyRange( :jeMatkou :Osoba )
```

```
ObjectPropertyDomain( :jeOtcom :Muž )
```

```
ObjectPropertyRange( :jeOtcom :Osoba )
```

```
DataPropertyDomain( :máVek :Osoba )
```

```
DataPropertyRange( :máVek xsd:nonNegativeInteger )
```

Ďalej vieme definovať inverznosť vlastností. Napríklad pokiaľ máme vlastnosť je rodičom, jej inverzná vlastnosť je je dieťaťom.

```
InverseObjectProperties( :jeRodičom :jeDieťaťom )
```

```
InverseObjectProperties( :jeMatkou :máMatku )
```

```
InverseObjectProperties( :jeOtcom :máOtca )
```

Niektoré vlastnosti by sme nechceli definovať iba jednosmerne ale aj obojsmerne. Takouto vlastnosťou sa myslí symetrická vlastnosť.

```
SymmetricObjectProperty( :jePríbuzný )
```

Jazyk OWL poskytuje definície rôznych obmedzení. Ako prvou si spomenieme disjunktnosť jednotlivých tried, kde jednotlivé objekty nemôžu patriť do oboch tried zároveň.

```
DisjointClasses( :Muž :Žena )
```

Ďalším obmedzením je definovanie či je konkrétny objekt tým istým alebo iným objektom.

```
DifferentIndividuals( :Janka :Martin )
```

```
DifferentIndividuals( :Janka :Jozef )
```

```
SameIndividual( :Martin :Maťo )
```

Taktiež vieme definovať objekty, ktoré spadajú pod rovnakú triedu, pokiaľ spĺňajú určité vlastnosti. Napríklad, Všetky ženy, ktoré sú zároveň rodičom, sú aj Matkou. Vieme povedať aj to, že všetky objekty, ktoré sú buď matkou alebo otcom, sú zároveň aj rodičom.

```
EquivalentClasses(
```

```
  :Matka
```

```

    ObjectIntersectionOf( :Žena :Rodič )
)

```

```

EquivalentClasses(
    :Rodič
    ObjectUnionOf( :Matka :Otec )
)

```

Ďalej vieme definovať existenčné obmedzenia na objekt a vlastnosti. Napríklad, každý rodič musí mať dieťa.

```

EquivalentClasses(
    :Rodič
    ObjectSomeValuesFrom( :jeRodičom :Osoba )
)

```

Podobne ako existenčné si vieme definovať aj všeobecné obmedzenie. Napríklad rodiča, ktorý ma iba chlapcov si zdefinujeme nasledovne:

```

EquivalentClasses(
    :RodičIbaChlapci
    ObjectAllValuesFrom( :jeRodičom :Muž )
)

```

V príklade vyššie nám ale toto obmedzenie nezaručuje, že daný rodič musí mať dieťa. Na takéto zabezpečenie sa využíva existenčné a všeobecné obmedzenie zároveň. Týmto obmedzením hovoríme že musí mať minimálne jedno dieťa, ktoré je chlapcom a zároveň všetky jeho deti sú chlapci.

```

EquivalentClasses(
    :RodičIbaChlapci

```

```
ObjectIntersectionOf(  
  ObjectAllValuesFrom( :jeRodičom :Muž )  
  ObjectSomeValuesFrom( :jeRodičom :Muž )  
)  
)
```

MH: Kedze cela Tvoja praca je o ontologiach a budes ich zrejme nejako zapisovat mozno by si mohol nieco povedat aj o jazykoch na zaspis ontologii, minimalen apson o jednom, s ktorym budes pracovat dalej (cize zrejme OWL)

Kapitola 4

Existujúce ontologické riešenia

Množstvo kyber útokov v dnešnej dobe narastá závratnou rýchlosťou, čo značí že dnešné spôsoby a metódy ochrany nie sú dostatočné a preto je potrebné sa zamyslieť nad novými spôsobmi ochrany. Jeden z prístupov by mohol byť založený práve na ontológiách. Ontológie a systémy postavené nad nimi majú výhodu sémantiky, ktorá je schopná rozlišovať situácie kedy je počítačový systém normálny alebo škodlivý.

Problém s ktorým sa ale potýkame je ten, že neexistuje jednotný formát zápisu údajov. Väčšina nástrojov ktoré v dnešnej dobe existujú, majú vlastné štandardy. Keďže tieto štandardy sú prevažne rozdielne, nedá sa ich prepájať a využívať efektívne. Tento problém by mohol byť taktiež vyriešený vďaka ontologickému riešeniu. Tým pádom by sme vedeli mať také dáta, ktoré dokážu stroje nielen prečítať, ale zároveň aj pochopiť.

Ontologický prístup taktiež poskytuje jednoduchšiu rozšíriteľnosť už existujúcej ontológie a tým sa dá vytvárať presnejší popis záznamov.

Vďaka URI reprezentáciám jednotlivých entít, ktoré sú používané ako identifikátory jednotlivých objektov, nemôže nastať problém nepochopenia dát ako k tomu môže dochádzať v ľudskej reči. Napríklad ak by sme

povedali slovo *koruna*, nikto nevie, či máme na mysli korunu stromov alebo kráľovskú korunu. Avšak vďaka atribútom vieme toto slovo lepšie pochoiť, keďže nám ho atribúty bližšie definujú.

MH: ↑ Tento text sa skor hodi do uvodu, pripadne cast do casti kde opisujes ontologie. Tuto kapitolu by som ocakaval, ze otvoris nejak zhruba odtialto ↓ (Akoze toto ↑ sa uz citatel (mal) dovediet niekde predtym a je zbytocne to tu opakovat...)

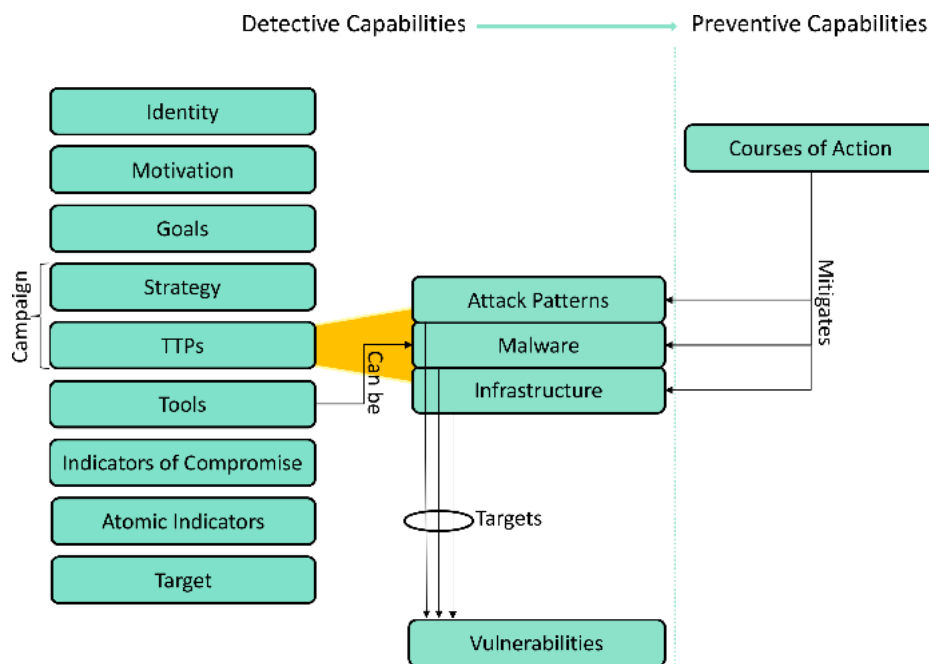
V súčasnosti existuje veľa rôznych štandardov a ontologických riešení pre doménu kyber bezpečnosti, avšak veľa z nich už ani nie je vyvíjaných. Organizácie, ktoré vyvíjali tieto ontológie, buď stratili o ďalší vývoj záujem, alebo už len nezverejňujú svoje pokroky v danej doméne, teda prešli na closed-source systém.

V nasledujúcich kapitolách si povieme niečo o zaužívaných štandardoch, základnom modeli, z ktorého vychádzame a podľa, ktorého posudzujeme, či je daná ontológia dobrá. Taktiež rozoberieme existujúce riešenia v doméne kyber bezpečnosti.

4.1 CTI model

Cyber Threat Intelligence model [MB17] (CTI model) má za cieľ objasňovať rôzne typy informácií, ktoré potrebujú organizácie zhromažďovať o kybernetických hrozbách. Cieľom modelu je jednotne definovať a podrobne popísať okruhy v danej doméne. Výhodou takéhoto modelu je zlepšenie schopností pri prevencii, detekcii a reakcii na bezpečnostné hrozby.

Tento model bol vyvinutý so zámerom, aby slúžil ako predloha pre ontologické riešenia, nakoľko autori zahrnuli fakt, že ontologické riešenia sú najlepšie pri publikovaní a zdieľaní dát a sú strojovo čitateľné vďaka typu reprezentácie.



Obr. 4.1: CTI model.

Zdroj: [MB17]

Model zahŕňa informácie o autorovi útoku, motivácie a ciele útoku, stratégiu akou je útok vedený, aké technológie a postupy používa, ktorým základným vzorcom útoku sa riadi, aké slabiny jednotlivé útoky využili, aké atomické indikátory dokážu predpovedať útok a akú cieľovú skupinu útok postihuje.

V nasledujúcich častiach popíšeme niektoré okruhy z obr. 4.1, ktorý znázorňuje jednotlivé prvky modelu.

4.1.1 Identita

Identita môže byť definovaná rôznymi typmi identifikátorov. Ak sa pozná presná totožnosť osoby alebo organizácie, ktorá útok iniciuje, identifikátorom je názov osoby alebo spoločnosti.

Pokiaľ takýto identifikátor nevieme určiť, pridelujeme im anonymnú identitu, ktorú následne vieme spájať vyhodnotením útoku. Pokiaľ zoberieme do úvahy všetky ostatné časti CTI modelu a vyhodnotíme zhodujúce sa časti, vieme povedať, za ktoré útoky zodpovedá rovnaká osoba alebo organizácia.

4.1.2 Kampaň

Kampaň popisuje samotný útok z dvoch pohľadov. Netechnický, ktorý je reprezentovaný stratégiou a technický, ktorý definuje taktiky, techniky a procedúry využívané v útoku.

Stratégia

Stratégia predstavuje netechnický popis útoku na hornej úrovni. Pokiaľ si zoberieme nejakého útočníka, tak tento útočník zvyčajne dosahuje ciele útoku viacerými spôsobmi a stratégia definuje, čo a ako je potrebné pozorovať.

4.1.3 Zraniteľnosť

Zraniteľnosti definujú slabiny systému. Tieto slabiny môžu byť definované rôznymi stavmi zariadenia ako napríklad, ktoré systémy bežia, ale môžu predstavovať aj nastavenia rôznych aplikácií, napríklad rôzne prístupové nastavenia a povolenia pre webové prehliadače. Taktiež môžu popisovať rôzne funkcie, ktoré umožňujú vykonávať jednotlivé systémy, ako napríklad spúšťanie rôznych príkazov ktoré ovplyvňujú a menia systémové nastavenia zariadenie.

TTP

Taktiky, techniky a procedúry (TTP) sa zamieravajú na dáta čitateľné strojmi. Popisujú spôsoby útoku z hľadiska toho čo chcú dosiahnuť a ako to robia.

Vzorec útoku je jedným z hľadísk TTP. Definuje akým spôsobom je útok použitý. Napríklad sa tu nachádzajú rôzne úpravy názvov súborov, rôzne prenastavenie časovania a stavov zariadenia, atď.

Malware je taktiež hľadisko z TTP. Definuje typ softvéru, ktorý sa vkladá do systému s úmyslom poškodiť cieľ útoku z hľadiska dôvernosti, integrity alebo dostupnosti. Medzi typické malware patrí vírus, trójsky kôň, červ, metódy tajných vstupov, spyware atď.

Každý okruh z TTP využíva niektoré zraniteľnosti systémov.

4.1.4 Indikátor

Indikátory predstavujú množinu rôznych vlastností zariadenia, kedy je zariadenie náchylné na útok. Tieto indikátory popisujú rôzne nastavenia zariadenia, jeho technické parametre alebo nainštalovaný softvér a vďaka nemu sú schopné definovať rôzne útoky.

Atomické indikátory sú najpremenlivejšie ukazovatele, nakoľko sa po čase menia. Napríklad ak niektoré útoky prebiehajú z určitej IP adresy, tak táto adresa sa môže časom zmeniť. Tieto typy dát môžu pomáhať pri identifikovaní útočníka iba v určitom čase.

4.1.5 Nástroj

Nástroje sú pomôcky, ktoré útočníci inštalujú v zariadení obete. Väčšinou zahŕňa špecializovaný softvér priamo určený na spôsobenie škôd, ako

napríklad vzdialené vykonávanie procesov alebo sieťové skenovanie. Taktiež môže byť použitý pre zabránenie detekcie útoku inými systémami.

4.1.6 Zámer

Zámer pomenováva koncový stav objektu útoku. Zámer nemusí byť vopred daný alebo zrejmý. Jednotlivé útoky môžu prebiehať aj s tým, že až počas útoku je zámer známy. Zámerom môže byť napríklad ukradnutie duševného vlastníctva, poškodenie systému, získanie kompromitujúcich dát a podobne.

4.1.7 Cieľ

Cieľom je myšlená entita, na ktorú je útok namierený. Takouto entitou môže byť napríklad organizácia, štát alebo konkrétna osoba.

MH: Ano, určite

MR: zahrnute, plus som sem vypichol casti ktore budeme popisovat z modelu a na ktore budem vyhodnocovat UCO podrobne

4.2 Unified Cybersecurity Ontology

Unified Cybersecurity Ontology [SPF⁺16] alebo skrátené UCO je rozšírením pôvodného projektu Intrusion Detection System (IDS), ktorého tvorcom je rovnaká skupina. Spája viaceré bežne dostupné bezpečnostné štandardy používané v kybernetickej bezpečnosti. Prevažne pokrýva STIX, ktorý je najväčším a najkomplexnejším štandardom, pokrývajúcim najväčšiu časť kybernetickej bezpečnostnej domény ale taktiež pokrýva iné relevantné štandardy ako CVE4, CCE5, CVSS6, CAPEC7, CYBOX8, KillChain9 a STUCCO10.

Aj keď je STIX najkomplexnejším štandardom a zjednocuje všetky

informácie o kybernetických hrozbách, má tieto dáta uložené v XML súboroch, takže nepodporuje výhody inferencie v ontológiách, čo UCO poskytuje.

Okrem týchto štandardov obsahuje aj mapovanie na všeobecné databázy ako sú Google Knowledge Graph, DBPedia a Yago. Vďaka týmto mapovaniam je možné mať prístup k verejným databázam z rôznych domén záujmu.

Základnými triedami, využívanými v UCO sú:

- *Means* – Čo je zamýšľané daným útokom.
- *Consequences* – Dôsledky útoku.
- *Attack* – Typ útoku.
- *Attacker* – Kto je iniciátorom daného útoku.
- *Attack-Pattern* – Vzorec útoku, podľa ktorého je útok riadený.
- *Exploit* – K čomu útok slúži.
- *Exploit Target* – K čomu slúži cieľ alebo výsledok útoku.
- *Indicators* – Indikátor útoku.

Každá z týchto tried je mapovaná na už reálne existujúcu triedu v niektorom z vyššie uvedených štandardov, prevažne na STIX schému.

Ontológia UCO umožňuje analytikom zachytávať špecifické vedomosti o kybernetickej bezpečnosti pomocou termínov a tried z ontológie a taktiež umožňuje písať pravidlá, ktoré sa môžu použiť na odvodenie nových poznatkov.

Vývojári extrahovali dáta z National Vulnerability Database (NVD), ktorá je uložená v XML súboroch. Potom boli namapované na triple store

DBPedia a dáta boli uložené na FUSEKI server, ktorý podporuje dopytovanie z rôznych zdrojov rovnako ako ich odvodzovanie.

MH: ↑ Zmienky o nejakých datach sa tu zjavajú z čista-jasna... Doteraz sme hovorili stále o ontológii, nezapadá to... (Mozno treba len previazať a upresniť?)

4.3 Integrated Cyber Analysis System

Integrated Cyber Analysis System[SW15] alebo ICAS je ontológia vytvorená pre TAPIO (Targeted Attack Premonition using Integrated Operational data) nástroj, ktorý je schopný extrahovať dáta z počítačov v jednej sieti do jedného sémantického grafu a tým zjednoduší a urýchli prácu bezpečnostným tímom pri vyhľadávaní ohrozenia systému, čím by sa zvýšila prehľadnosť dát a tiež znížil dopad útoku.

Samotná ontológia ICAS je veľmi komplexnou, nakoľko obsahuje približne 30 podontológií, kde každá sa špecializuje na inú oblasť v doméne informačnej bezpečnosti.

MH: ↑ vieme povedať, že či nejaká časť z toho je relevantná pre nás

Nástroj TAPIO spolu s ontológiou ICAS bol vyvíjaný organizáciou DAPRA, ale dátum poslednej úpravy bol v roku 2017, čiže podobne ako UCO sa jedná o projekt ktorý už nie je aktuálny.

MH: ↑ Tu by som bol opatrnejší, tvrdiť, že niečo nie je aktuálne, lebo posledný update bol pred 3 rokmi je zvláštne. Čo keď pred tromi rokmi to dotiahli už do dokonalosti a teraz už len používajú?

4.4 STUCCO

STUCCO [IBN⁺15], ktorej autorom je Iannacone at al. je ontológiou, ktorá je určená na prácu so znalostnými grafovými databázami. Jej základ tvoria

scenáre použitia ľudskými používateľmi alebo automatizovanými strojmi. Obsahuje dáta z 13 rôznych štruktúr, ktoré majú rôzne formáty a ktoré sú uložené v rôznych typoch databáz.

STUCCO obsahuje dáta z nasledovných kategórií do ktorých je rozdelená bezpečnostná doména.

- Identita – predstavuje totožnosť a vlastnosti útočníka.
- Taktika technika a procedúry (TPP) – Popisuje čo daný útok robí a ako to robí.
- Nástroje – Aké nástroje sú potrebné pre úspešné vykonaie útoku.
- Atomické indikátory – Sem môžu spadať súbory, IP adresy, doménové mená atď. Nanešťastie tieto dáta majú krátku životnosť nakoľko sa stále menia.

Časť II

Vlastný prínos

Kapitola 5

Výskum a Analýza UCO

MH: Takže si to vlastne sem cele presnul a v Casti I už o tom nie je ani zmienka? Tu by to malo byť uvedené spôsobom, že tu ju podrobne píšeme a analyzujeme... Cize malo by to byť v Casti II vedene ako analýza UCO, nie jej len popis

MR: Done

MH: Namiesto ↓ „popíšeme a budeme porovnávať“ napíše „podrobne analyzujeme a vyhodnocujeme vzhľadom na“.. Nech je to ešte jasnejšie.

MR: Done

MH: Len už v prvej vete ide o UCO? Ano, nie je to jasné, keď je tam len citácia.

Tiež by tu mohol byť jej názov, nie len skratka. Dotreť: pozor na tie merzery okolo zatvoriek, aj hranatých, teda aj citácii...

V tejto časti podrobne analyzujeme a vyhodnocujeme ontológiu[Lab17], ktorú sme si vybrali pre ďalší vývoj a budeme vyhodnocovať UCO ontológiu vzhľadom na CTI model. Pre túto prácu sú pre nás najpodstatnejšie časti Identita, Útok, Kampaň (Stratégia a TTP), Slabina, Indikátor a Nástroj. Na tieto oblasti sa zameriame v nasledujúcich kapitolách, vyhodnotíme, či sú v ontológii popísané dostatočne, či sú prepájané s už existujúcimi ontológiami a či sú ich vlastnosti a hierarchia dostatočné.

5.1 Model

Základnou triedou tejto ontológie je *UCO Thing*, ktorá je nadrtiedou každej triedy v ontológii.

Pokiaľ UCO ontológia využíva už existujúcu triedu z inej ontológie, zadefinuje si ju aj pre svoju doménu a pomocou jazyka OWL jej zadefinuje ekvivalentnú triedu, čím zabezpečí opätovné použitie dát pre iný zdroj. Napríklad trieda *Indicator* je vďaka tomuto vzťahu prepojená s ontológiou CAPEC.

Identita je v ontológii UCO reprezentovaná triedou *Attacker*, ďalej útočník, ktorý je namapovaný na existujúcu triedu *ThreatActor* zo STIX-u.

MH: Možno by sme mohli na obrázku tie triedy z iných ontológií nejako lepšie vyznačiť? Je blbosť vyrobiť väčší obdĺžnikovitý biely tvar s nadpisom STIX zapuzdrujúci veci zo STIXU (hoc by tam bola aj len 1 trieda)?

Každý útočník má nejaké meno (*hasTitle*) a je priradený k existujúcim incidentom reprezentovaných triedou *Incident* priradených vlastnosťou *hasRelatedIncident*. Má aj určitú mieru doveryhodnosti (*hasConfidenceType*). Každý útočník má priradené nejaké zavedené postupy a stratégie pri útoku, ktoré sú inštanciou triedy *Campaign* priradené vlastnosťou *hasAssociatedCampaign*.

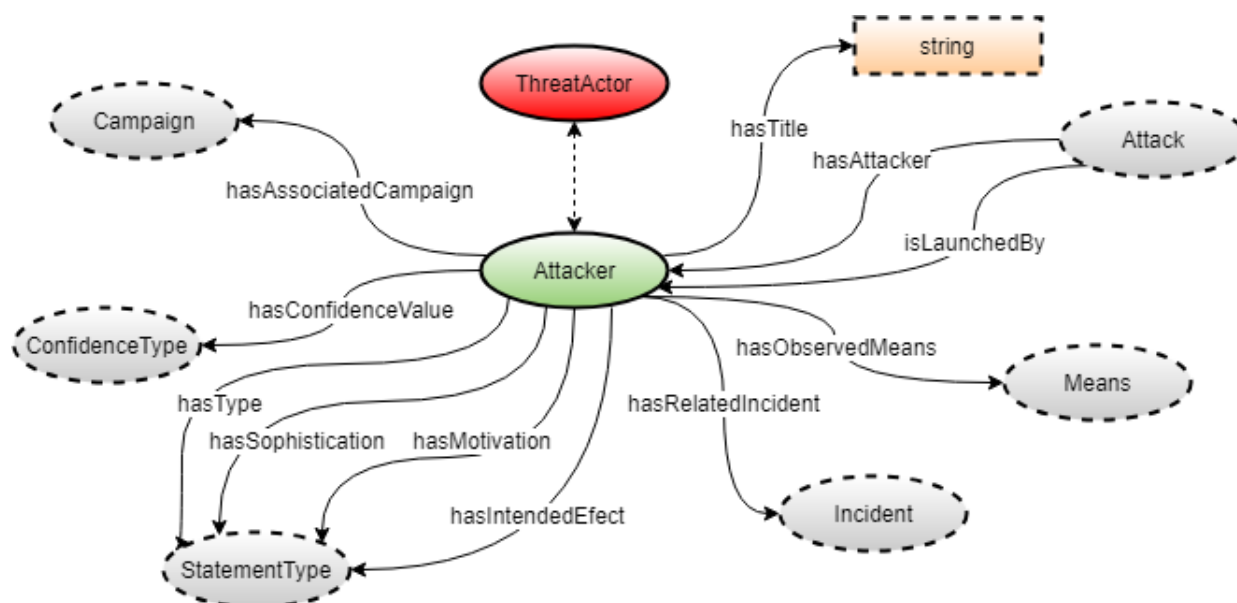
MH: ↑ chyba premostenie na tabuľku vlastností, možno aj viac ich popísať v texte (?)

Útok máme reprezentovaný triedou *Attack*, ktorá má viaceré pravidlá na splnenie. Musí mať minimálne jeden význam (*Means*) a minimálne jeden následok (*Consequence*) útoku. Samotný význam je ekvivalentný s triedou TTP, ktorá popisuje samotný útok pomocou informácií o vzoroch útoku, využívaní slabín systémov alebo známych malwaroch. Nakoľko vďaka jazyku OWL vieme definovať napríklad inverzné vlastnosti je žiadúce aby sa táto vlastnosť využívala. Pre vlastnosť *hasAttacker* to žiaľ nie je zadefinované a

Identita – Attacker		
Vlastnosť	Doména	Rozsah
hasAssociatedCampaign	Attacker	Campaign
hasConfidenceValue	Attacker	ConfidenceType
hasIntendedEffect	Attacker	StatementType
hasMotivation	Attacker	StatementType
hasObservedMeans	Attacker	Means
hasRelatedIncidents	Attacker	Incident
hasSophistication	Attacker	StatementType
hasType	Attacker	StatementType
hasTitle	Attacker	xsd:string

Tabuľka 5.1: Tabuľka vlastností popisujúcej časť **Identita** z CTI modelu.

určite by to bolo vhodné zadefinovať.



Obr. 5.1: UCO ontológia – časť Identita

MH: ↑ Tu zasa strasne skaces, viac vysvetluj... Kde sa vzalo TTP? Tiez pises, ze *hasAttacker* nie je definovane, ale v tabulke to je...

MR: tym som povedal ze nie je zadefinovana ziadna inverzna vlastnost k vlastnosti *hasAttacker*

MH: Ano, ale je to tu take vlepane, citatel nechape, preco prave tu uvadas takuto poznaku, navyse, asi sa to tyka aj inych vlasnosti, ku ktorym by sa dala dodefinovat inverzna (?) A dalsia vec, nadvazujeme niekde na to? Navrhujeme vylepsenia tohto druhu?

Útok – Attack		
Vlastnosť	Doména	Rozsah
<i>hasAttacker</i>	Attack	Attacker
<i>hasConfidenceValue</i>	Attack	ConfidenceType
<i>hasIndicator</i>	Attack	Indicator
<i>hasMeans</i>	Attack	Means
<i>hasObservable</i>	Attack	Observable
<i>hasRequestedCOA</i>	Attack	CourseofAction
<i>hasSource</i>	Attack	Source
<i>hasTakenCOA</i>	Attack	CourseofAction
<i>isLaunchedBy</i>	Attack	Attacker

Tabuľka 5.2: Tabuľka vlastností popisujúcej časť **Útok** z CTI modelu.

Kampaň je zahrnutá v triede *Campaign*, ktorá môže existovať iba v prípade, že už bola použitá v nejakom útoku. Každá kampaň môže taktiež mať nejakú ďalšiu kampaň, ktoré medzi sebou súvisia, kde táto vlastnosť je obojsmerná. Taktiež má vlastnosť *hasCampaign*, ktorej doménou je trieda *Indicator*. Táto trieda pochádza z ontológie CAPEC, ktorá obsahuje známe vzory útokov.

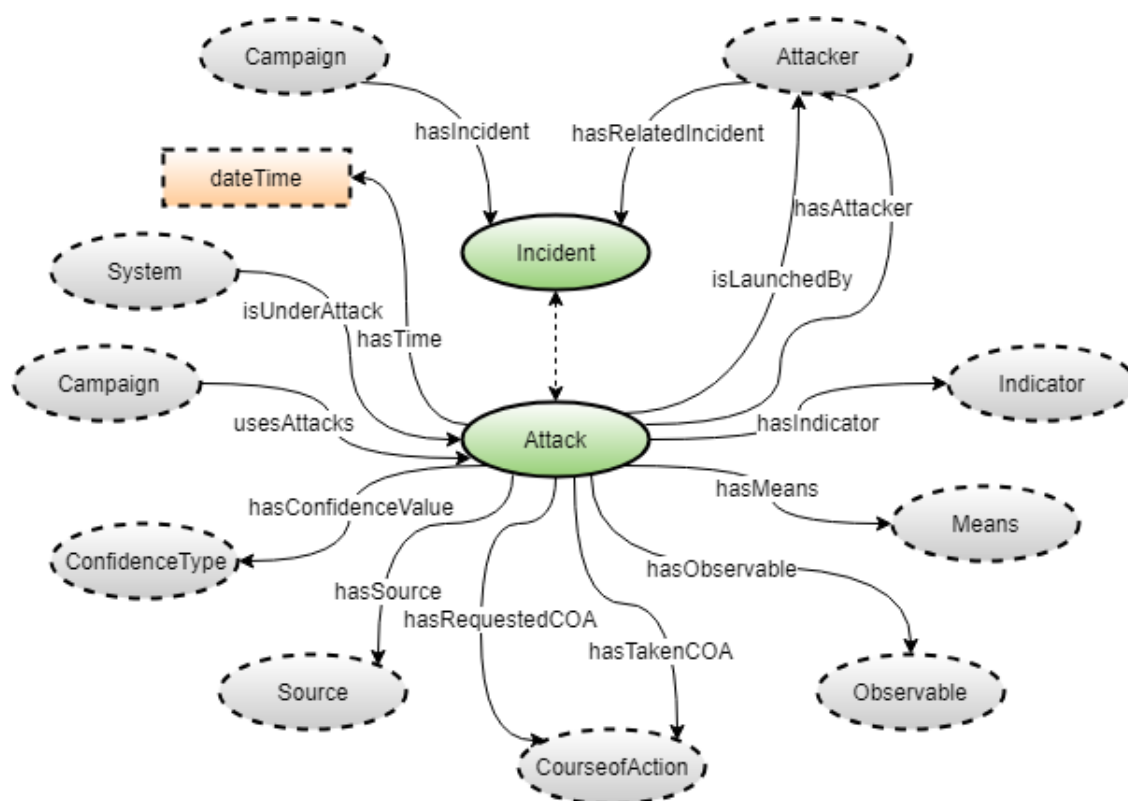
MH: ↑ Tu (ale aj inde) je taky problem, ze v tabulke je mnozstvo vlastnosti triedy Attack, ci Campaign, ale v texte vysvetlujes len niektore. Nemusis vsetky vymenovavat, ale mozno by sa dalo aspon zhrnut, zgurpit ich... Nieco ako: Ostatne vlasntosti triedy Attak definuju/umoznuju priradit/... ...

Slabiny sú zahrnuté v triede *Vulnerability*, ktorá nie je namapovaná na

Kampaň – Campaign		
Vlastnosť	Doména	Rozsah
hasAssociatedCampaign	Campaign	Campaign
hasCampaign	Indicator	Campaign
hasIndicator	Campaign	Indicator
hasIncident	Campaign	Incident
hasMenas	Campaign	Means
hasStatus	Campaign	
isLaunchedBy	Campaign	Attacker
usesAttacks	Campaign	Attack

Tabuľka 5.3: Tabuľka vlastností popisujúcej časť **Kampaň** z CTI modelu.

žiadnu existujúcu ontológiu, avšak dáta už existujú v rámci databázy CVE, ktorá predstavuje dáta o softvérových a hardvérových chybách. Tieto



Obr. 5.2: UCO ontológia – časť Útok

slabiny sú naviazané na objekty typu *Product* vlastnosťou *affectsProduct*, ktoré môžu byť buď hardvérové alebo softvérové a každá slabina ovplyvňuje niektorý produkt. Taktiež je v nej zaznamenaný čas objavenia (*discoveryTime*), spôsob narušenia alebo preniknutia do systému (*hasAccessVector*), zložitosť preniknutia do systému (*hasAccessComplexity*), rôzne typy dopadov alebo skóre úspešnosti (*score*). Môže niesť aj informáciu o zdroji slabiny. Taktiež definuje objekty, ktoré je potrebné pozorovať, ktoré sú typu *Observable*. Táto trieda zatiaľ nie je namapovaná na žiadnu existujúcu, ale vieme že existuje ontológia CyBox, ktorá tieto dáta uchováva.

Slabina – Vulnerability		
Vlastnosť	Doména	Rozsah
<i>affectsProduct</i>	Vulnerability	Product
<i>discoveryTime</i>	Vulnerability	xsd:dateTime
<i>exploitsVulnerability</i>	Means	Vulnerability
<i>hasAccessComplexity</i>	Vulnerability	xsd:string
<i>hasAccessVector</i>	Vulnerability	xsd:string
<i>hasConsequences</i>	Vulnerability	Consequence
<i>hasCVE_ID</i>	Vulnerability	CVE
<i>hasMeans</i>	Vulnerability	Means
<i>hasObservable</i>	Vulnerability	Observable
<i>publishedDateTime</i>	Vulnerability	xsd:dateTime
<i>score</i>	Vulnerability	xsd:float

Tabuľka 5.4: Tabuľka vlastností popisujúcej časť **Slabina** z CTI modelu.

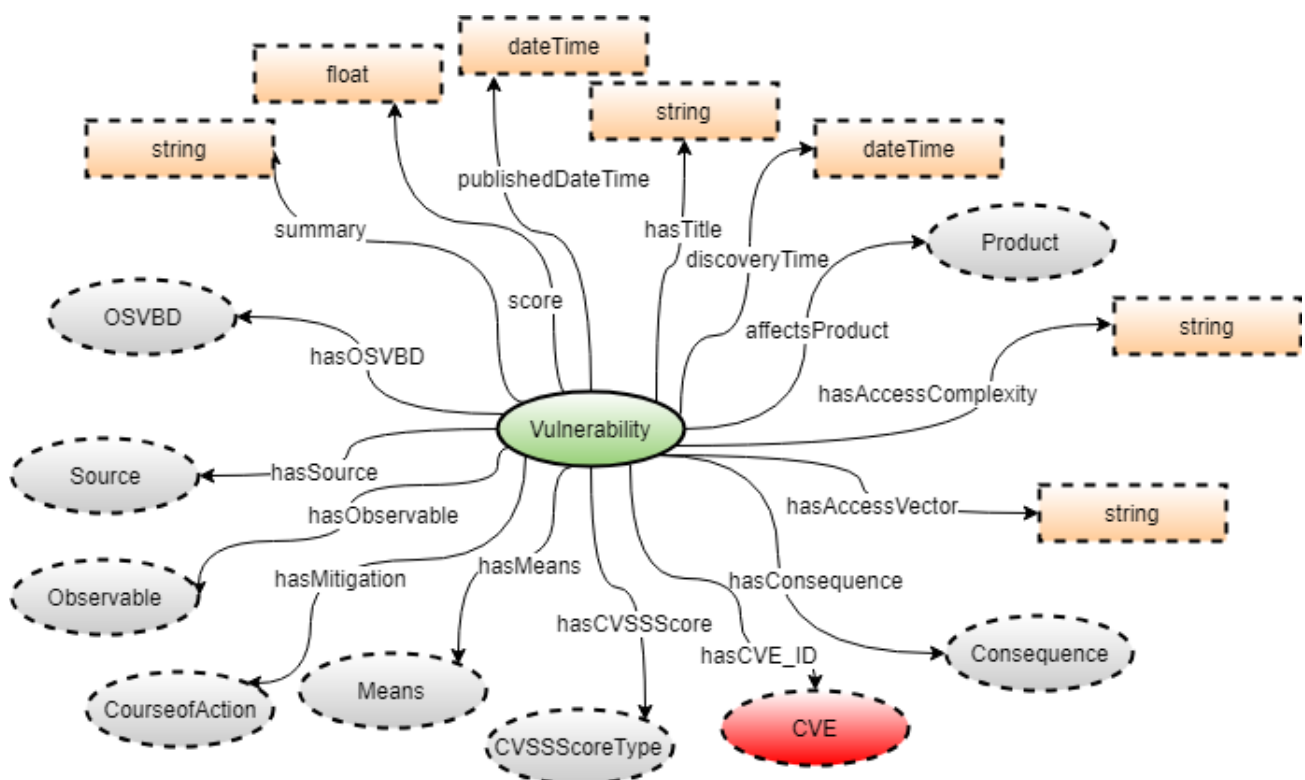
Taktiež existuje trieda *CWE*, ktorá je prepojená s triedou *Weakness* z už existujúcej CWE databázy. Taktiež ako trieda *Vulnerability*, popisuje slabiny z CTI modelu. Trieda *CWE* popisuje známe typy slabostí hardvérov a softvérov. Obsahuje informáciu o čase zistenia (*timeOfIntroduction*) a stručný popis (*description*). Zvyšné vlastnosti nemajú definované rozsahy, čo je výrazným nedostatkom a je potrebné tieto vlastnosti dodefinovať.

MH: ↑ V tabulke 5.4 však ale všetky vlastnosti majú aj rozsah, o ktoré zvyšne teda ide?

Slabina – CWE		
Vlastnosť	Doména	Rozsah
timeOfIntroduction	CWE	xsd:dateTime
discoveryTime	CWE	xsd:dateTime
commonConsequences	CWE	Consequences

Tabuľka 5.5: Tabuľka vlastností popisujúcej časť **Slabina** z CTI modelu.

Indikátory sú reprezentované triedou *Indicator*. Táto trieda je namapovaná na už existujúcu triedu z ontológie CAPEC. Indikátory spadajú pod kampane (*hasCampaign*) a majú určitý dopad (*hasImpact*).



Obr. 5.3: UCO ontológia – časť Slabiny

MH: CVE -> CWE?

Každý indikátor má aj význam (*hasMeans*) a objekty na pozorovanie z triedy *Observable* (*hasObservable*). Každý indikátor môže mať nejaký iný indikátor, ktorý s ním súvisí (*hasRelatedIndicator*). Obsahuje aj navrhovaný postup reprezentovaný triedou *CourseOfAction* (*hasSuggestedCOA*).

Indikátory – Indicator		
Vlastnosť	Doména	Rozsah
hasCampaign	Indicator	Campaign
hasConfidenceValue	Indicator	ConfidenceType
hasImpact	Indicator	StatementType
hasIndicator	CWE	Indicator
hasKillChainPhase	Indicator	KillChainPhase
hasMeans	Indicator	Means

Tabuľka 5.6: Tabuľka vlastností popisujúcej časť **Indikátor** z CTI modelu.

Nástroje TODO – chcem sa pobaviť s panom baloghom čo by radil medzi ne z uco ontologie, nakoľko sa nevidím len do tejto domény aby som vedel povedať ktorá do toho spada.

Kapitola 6

Návrh ontológií

V predchádzajúcej časti sme si podrobnejšie predstavili vybranú časť UCO ontológie. Ukázali sme si, aké časti z CTI modelu spĺňa a aké dáta jednotlivé okruhy reprezentujú. Touto podrobnejšou anlyzou sme zistili, ktoré dáta ešte nepokrývajú alebo sa nemapujú na už existujúce dáta.

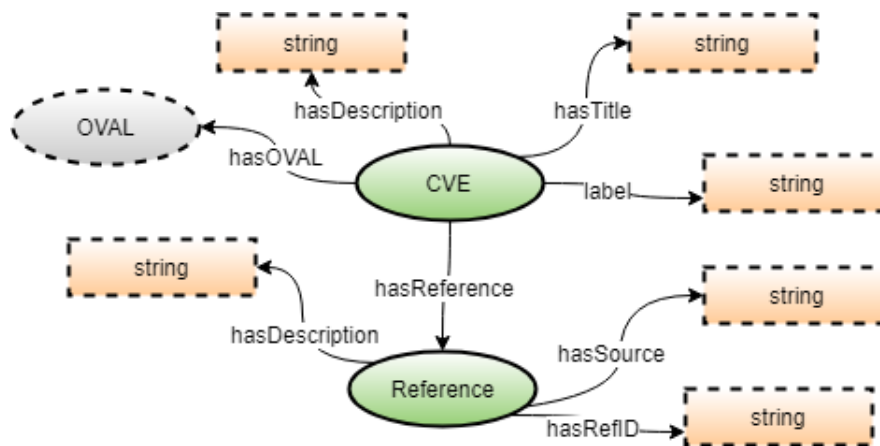
V tejto kapitole si ukážeme databázy CVE a OVAL, ktoré predstavujú z CTI modelu časť Zraniteľnosť. Predstavíme si nami vytvorené ontológie, ich mapovania a vlastnosti jednotlivých objektov.

6.1 CVE

Common Vulnerabilities and Exposures(CVE) je databáza všetkých chýb zabezpečenia v oblasti kybernetickej bezpečnosti. Jej úlohou je identifikovať, definovať a katagolizovať tieto chyby, ďalej zraniteľnosti. Každá zraniteľnosť má vlastný záznam, ktorý prechádza fázami v databáze. Zraniteľnosť je najprv objavená, potom pridelená a nakoniec zverejnená. Zverejňovať tieto zraniteľnosti môžu iba tí, ktorí majú partnerstvo s programom CVE. Tieto organizácie dostanú pridelenú množinu CVE

záznamov pre daný rok, ktorú môžu postupne naplňať rôznymi zraniteľnosťami, ktoré objavia. CVE záznamy sú následne používané pri komunikácii a riešení rôznych bezpečnostných hrozieb, aby sa zabezpečilo rovnaké chápanie problému, ktoré bolo konzistentne definované v CVE.

Tieto záznamy majú svoje identifikátory, ktoré sa skladajú z prefixu *CVE*, následne obsahujú rok objavenia a nakoniec unikátny identifikátor v rámci tejto trojkombinácie oddelenej pomlčkami. Napríklad identifikátor *CVE-2020-0035* predstavuje zraniteľnosť z roku 2020 v databáze CVE s identifikátorom v danom roku 0035. Podľa množstva záznamov za daný rok, sa odvíja počet cifier na definovanie identifikátora zraniteľnosti. Napríklad v roku 1999 kedy sa začala databáza CVE naplňať, dosahoval počet zraniteľností ročne 1 600 záznamov, kdežto za rok 2020 to bolo 36 282 záznamov.



Obr. 6.1: CVE ontológia

Každý jeden záznam má taktiež popis, ktorý musí obsahovať dostatok informácií o tom, na ktoré systémy má daná zraniteľnosť vplyv, pokiaľ sa vzťahuje aj na konkrétnu verziu, tak aj táto verzia by mala byť uvedená. Musí obsahovať aj informácie o type zraniteľnosti, jej príčinu a dopad. Daný

popis musí byť napísaný v angličtine.

Záznamy majú aj zoznam referencií, ktoré sú používané ako externé zdroje k zraniteľnosti. Sú to externé odkazy s informáciou o zdrojovom systéme. Tieto odkazy musia používať iba protokoly *http*, *ftp*, *https*, alebo *ftps*.

Na základe týchto informácií sme vytvorili ontológiu, ktorá obsahuje základné informácie zraniteľnosti, podľa požiadaviek štandardu. Ako definíciu zraniteľnosti, používame jej URI, ktorá poskytuje prepoužitie a odkazovanie sa na ňu. Táto zraniteľnosť má taktiež svoj názov, ktorý reprezentujeme ako CVE identifikátor. Každá zraniteľnosť má svoj popis a zoznam referencií. Ako rozšírenie sme použili vlastnosť *hasRefId*, ktorá definuje unikátny identifikátor, v referencovanom systéme. Tento parameter nie je povinný.

6.2 OVAL

Open Vulnerability and Assessment Language(OVAL) je štandard, ktorý umožňuje popísanie zraniteľností jednotlivých produktov (softvérov). Tieto informácie sú písané vo formáte XML. Obsahuje tri hlavné kroky posudzovania.

Prvým krokom sú základné informácie o systémoch, ako napríklad o aký operačný systém sa jedná, ktoré jeho verzie ohrozuje a aké konkrétne produkty sú ohrozené, pokiaľ je splnená konfigurácia rôznych systémov. Taktiež obsahuje rôzne odkazy na ďalšie informácie o jednotlivých zraniteľnostiach.

Druhé posudzovanie analyzuje systém podľa špecifikovaného nastavenia a zisťuje či je daný systém ohrozený alebo nie.

Posledný krok tvorí správu o výsledku tejto analýzy. Na výstupe je informácia o aktuálnom konfiguračnom nastavení systému.

Pre našu ontológiu nie je potrebné reprezentovať všetky znalosti o OVAL-e. Vybrali sme si konkrétne časti, ktoré je potrebné publikovať v našich spravodajských dátach.

V ontológii reprezentujeme všeobecné informácie popisujúce a definujúce OVAL záznamy, ich vzťahy s jednotlivými operačnými systémami, ich verziami a produktami. Nakoľko vieme že jednotlivé operačné systémy majú rôzne verzie, vieme aj tieto dáta napävnno spájať. Zahrňame aj informácie o externých zdrojoch pre záznamy. Túto vlastnosť prepoužívame z vyššie definovanej časti **6.1 CVE**. Oproti CVE ale tieto dáta nemajú popis. Túto vlastnosť ale ponecháme, kôli možnému rozšíreniu. OVAL záznamami sa taktiež referencujú na rôzne CVE záznamy, ktoré podrobnejšie popisujú. V našej ontológii takýto typ referencie budeme popisovať špecifickou vlastnosťou.

Taktiež predpripravíme ontológiu na reprezentovanie kritérií, ktoré predstavujú logické testy. Tieto testy majú v sebe aj výrokovú logiku reprezentovanú pomocou logických operátorov AND a OR, kde je možné definovať viacúrovňové logické výrazy. Jednotlivé kritéria majú stručný popis a obširnejší popis, kde sú definované jednotlivými testami alebo ďalšími OVAL záznamami.

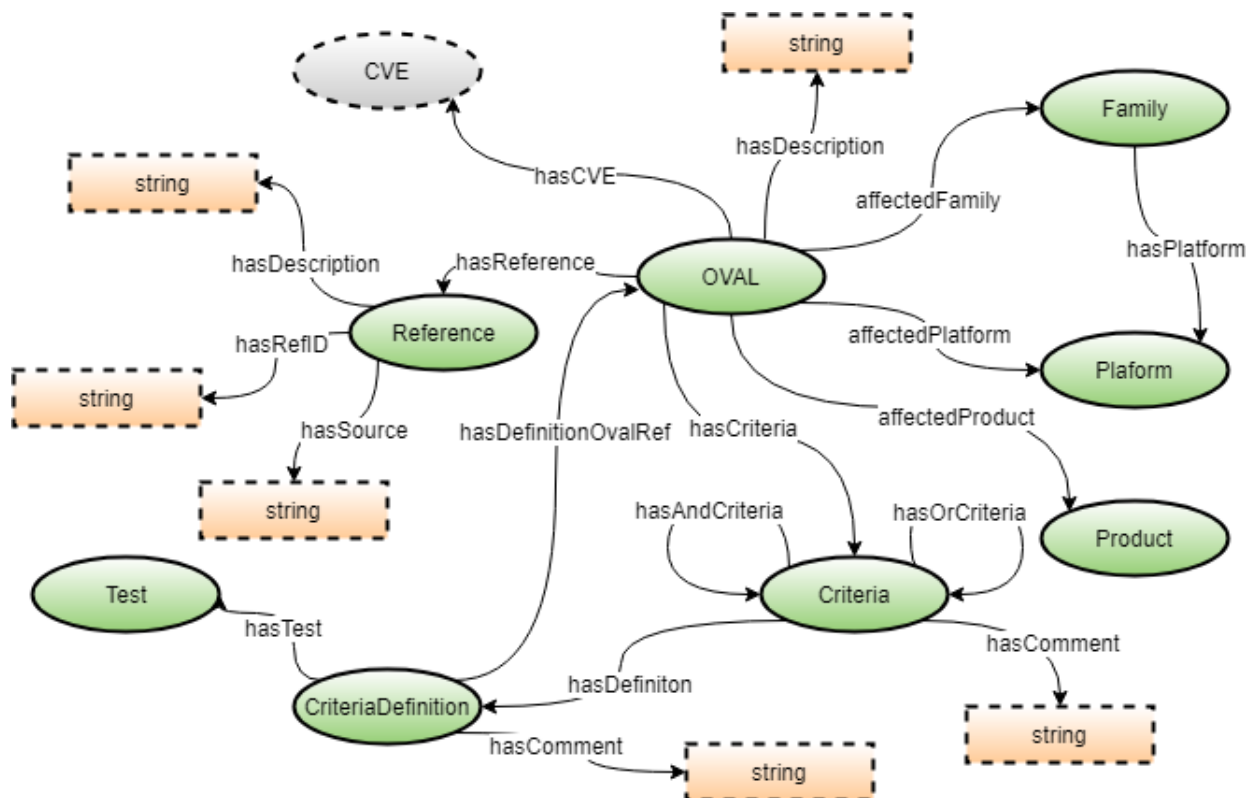
Každý OVAL objekt je reprezentovaný pomocou URN *oval:org.cisecurity:{def}:127* kde *{def}* musí byť z nasledujúcich možností:

- **def** – jedná sa o záznam definujúci zraniteľnosť
- **tst** – záznam, ktorý definuje test
- **var** – záznam definujúci množinu verzií platformy alebo produktu

- **obj** – príkaz, ktorý je potrebný spustiť v konzole
- **ste** – objekt, popisujúci výstup z konzoly s definovanou operáciou na kontrolu ako je regulárny výraz, porovnanie alebo výber z množiny verzií

Samotný záznam sa skladá z:

- *hasTitle* – názov záznamu
- *hasDescription* – popis záznamu
- *affectedFamily* – operačný systém, ktorý ovplyvňuje
- *affectedPlatform* – akú platformu operačného systému ovplyvňuje



Obr. 6.2: OVAL ontológia

- *affectedProduct* – ktorý produkt ovplyvňuje
- *hasReference* – odkaz na externý zdroj informácií súvisiaci so záznamom
- *hasCriteria* – zoznam logických testov potrebných na zistenie či daný záznam ovplyvňuje systém
- *hasCVE* – CVE objekt, ktorý popisuje daný OVAL záznam

Kritériá popisujúce test majú nasledujúce vlastnosti:

- *hasAndCriteria* – logické pravidlo spojené s rodičom AND operátorom
- *hasOrCriteria* – logické pravidlo spojené s rodičom OR operátorom
- *hasDefinition* – definícia kritéria
- *hasComment* – popis kritéria alebo definície kritéria
- *hasTest* – definícia vykonania a kontroly výstupu z testu

Kapitola 7

Implementácia a testovanie

V kapitole Návrh ontológií sme si predstavili nami vytvorené ontológie z už existujúcich databáz CVE a OVAL.

V tejto kapitole si ukážeme ako sme na tieto ontológie namapovali už existujúce dáta, ako sme tieto dáta migrovali a ako ich reprezentujeme na webe. Taktiež si ukážeme rôzne príklady použitia a analýzy nad importovanými dátami.

7.1 Dáta

Dáta, ktoré sme spracovávali pochádzajú z verejne dostupného zdroja štandardov OVAL a CVE. Tieto dáta mali pôvodne štruktúru XML, ktorá ale nebola písaná OWL syntaxov. Tieto dáta sme spracovali pomocou programovacieho jazyka Python. Jednotlivé triedy v nami navrhnutých ontológiách sme si implementovali a následne sme spracovali dané dátové súbory pomocou knižnice *xml*. Po sparsovaní dát sme tieto dáta definovali pomocou tripletov a následne ich zapísali pomocou knižnice *rdflib* do súboru vo formáte *turtle*.

Tento formát sme vybrali najmä kôli menšiemu objemu dát, nakoľko bolo potrebných viac optimalizácií pre generovanie. Ale aj po tejto optimalizácii sme nevedeli spracovať niektoré skupiny záznamov, preto sme sa rozhodli ich rozdeliť. CVE dáta sme spracovali tak, že sme ich rozdelili podľa rokov vytvorenia. OVAL dáta sme rozdelili podľa operačného systému, ktorého sa týkajú. Po zapísaní časti dát, sme túto časť premazali z pamäte a prešli sme na ďalšiu časť.

Nakoniec sme tieto migrované dáta nahrali na webovú službu github, ktorá poskytuje možnosť nahrávať zdrojové súbory do repozitára.

7.2 Testovanie

V predchádzajúcich častiach sme si popísali ako sme dáta migrovali a mapovali na nami vytvorenú ontológiu. Taktiež sme si popísali webovú stránku, ktorú sme vytvárali v spolupráci so študentami z FEI STU, ktorá slúži na zobrazovanie našich dát.

V tejto kapitole sa budeme venovať testovaniu našej vygenerovanej ontológie a dopytovanie sa nad nami importovanými dátami. Ukážeme si rôzne testy použiteľnosti a ukážky SPARQL dopytov.

7.2.1 Príklady použitia

Všetky nasledujúce príklady použitia boli overené na lokálne nainštalovanom triple store, do ktorého sme si naimportovali všetky migrované dáta. Každý jeden SPARQL dopyt sme sputili a na základe jeho výsledkov sme vyhodnocovali jednotlivé prípady použitia.

Príklad 1

Názov

CVE podľa platformy

Zadanie

Podľa zadanej platformy, nájdi v databáze všetky CVE objekty, ktoré súvisia s danou platformou. Na výstupe vráť ich URI definíciu s názvom CVE záznamu.

Dopyt

```
PREFIX main: <http://www.semanticweb.org/rycht/ontologies
/cyber_security_ontology#>
PREFIX platform: <http://www.semanticweb.org/rycht/ontologies
/cyber_security_ontology/platform#>
PREFIX cve: <https://cve.mitre.org/about/terminology.html#>
PREFIX oval:<https://oval.mitre.org/language/version5.11/OVAL>

SELECT ?cve ?title
WHERE {
    ?cve a cve:CVE.
    ?cve main:hasTitle ?title.
    ?oval a oval:.
    ?oval main:hasCVE ?cve.
    ?oval main:affectedPlatform platform:Some_platform
}
```


Výstup

Nad výstupom tohoto dopytu sme spravili analýzu, kde sme zistili, že najčastejšie vyskytujúca sa platforma mala priradených skoro 10 000 CVE záznamov z aktuálnych 204 144.

Príklad 2

Názov

OVAL záznamy, ktoré ovplyvňujú iba platformu.

Zadanie

Vráť mi všetky OVAL záznamy, ktoré ovplyvňujú všeobecne platformu a žiaden konkrétny produkt pre danú platformu. Na výstupe mi pošli ich URI OVAL záznamu, názov, URI a názov operačného systému a URI a názov platformy.

Dopyt

```
PREFIX main: <http://www.semanticweb.org/rycht/ontologies
/cyber_security_ontology#>
PREFIX cve: <https://cve.mitre.org/about/terminology.html#>
PREFIX oval:<https://oval.mitre.org/language/version5.11/OVAL>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?oval ?title ?family ?familyName ?platform ?platformName
WHERE {
    ?oval a oval:.
    ?oval main:hasTitle ?title.
    ?oval main:affectedFamily ?family.
```

```
?family rdfs:label ?familyName.  
?oval main:affectedPlatform ?platform.  
?platform rdfs:label ?platformName.  
NOT EXISTS {  
    ?oval main:affectedProduct ?product.  
}  
}
```

Výstup

Záznamov, ktoré spĺňajú dané zadanie sme identifikovali z takmer 20 000 oval záznamoch 9 065. Tým sme získali určitú predstavu o počte OVAL záznamoch, ktoré všeobecne ovplyvňujú iba platformy operačných systémov.

Príklad 3

Názov

Frekvenčná tabuľka CVE záznamov.

Zadanie

Získaj všetky CVE záznamy, zoradené podľa toho, ako často sú využívané ako referencia v OVAL záznamoch. Zoraď ich podľa využívanosti od najväčšej a na výstup vráť URI a názov CVE záznamu s počtom OVAL definícií, v ktorých sú použité.

Dopyt

```
PREFIX main: <http://www.semanticweb.org/rycht/ontologies  
    /cyber_security_ontology#>  
PREFIX platform: <http://www.semanticweb.org/rycht/ontologies
```

```
/cyber_security_ontology/platform#>
PREFIX cve: <https://cve.mitre.org/about/terminology.html#>
PREFIX oval:<https://oval.mitre.org/language/version5.11/OVAL>

SELECT ?cve ?title (COUNT(?oval) as ?count)
WHERE {
    ?cve a cve:CVE.
    ?cve main:hasTitle ?title.
    ?oval a oval:.
    ?oval main:hasCVE ?cve.
}
GROUP BY ?cve ?title
ORDER BY DESC(?count)
```

Výstup

Najviac používaný CVE záznam v OVAL záznamoch bol odkazovaný 12 krát. Zvyšné záznamy boli vždy odkazované aspoň 1 krát, čiže sme zistili že každý CVE záznam je podrobnejšie popísaný v OVAL záznamoch. Neexistuje záznam, ktorým by nebol spracovaný v OVAL databáze.

Príklad 4

Názov

Cve záznamy ovplyvňujúce viac operačných systémov.

Zadanie

Nájdite také CVE záznamy, ktoré ovplyvňujú rovnaký produkt na rôznych operačných systémoch. Na výstup vráť URI a názov CVE záznamu, produkt

ktorý ovplyvňuje a URI operačných systémov.

Dopyt

```
PREFIX main: <http://www.semanticweb.org/rycht/ontologies
/cyber_security_ontology#>
PREFIX platform: <http://www.semanticweb.org/rycht/ontologies
/cyber_security_ontology/platform#>
PREFIX cve: <https://cve.mitre.org/about/terminology.html#>
PREFIX oval:<https://oval.mitre.org/language/version5.11/OVAL>
```

```
SELECT ?cve ?title ?product ?family1 ?family2
WHERE {
    ?cve a cve:CVE.
    ?cve main:hasTitle ?title.
    ?oval1 a oval:.
    ?oval1 main:hasCVE ?cve.
    ?oval1 main:affectedProduct ?product.
    ?oval1 main:affectedFamily ?family1.
    ?oval2 a oval:.
    ?oval2 main:hasCVE ?cve.
    ?oval2 main:affectedProduct ?product.
    ?oval2 main:affectedFamily ?family2.
    FILTER (?family1 != ?family2).
}
```

Výstup

Počet CVE záznamov, ktoré ovplyvňujú viac ako jeden operačný systém je z 204 144 CVE záznamov 146.

7.3 Internetová stránka

V spolupráci so študentami môjho konzultanta, sme vytvorili internetovú stránku, založenú na našej ontológii a našich vygenerovaných dátach. Táto stránka bude skúziť na zobrazenie CVE a OVAL dát, kde bude možné tieto dáta prezerať a vyhľadávať. Pomocou SPARQL dopytov sa vyhľadávajú jednotlivé dáta a ich konkrétne informácie.

Takiež sa v nej budú dať prezerať dáta o jednotlivých operačných systémoch, ich platformách a produktoch. Na týchto stránkach budú informácie o jednotlivých OVAL a CVE záznamoch, ktoré sa týkajú daného predmetu.

Základná stránka pre každý OVAL a CVE objekt bude zobrazovať všetky dáta, ktoré má v databáze definované na prvej úrovni t.j., dáta kde daný objekt figuruje ako predmet v trojici.

Kapitola 8

Záver

MH: ↓ V kapitole Zaver nie je toto zhrnutie uz nutne, lepsie by bolo tuto cast dat prec, a zrusit podnadpis 8.1 a zacat (bez podnadpisu) tym zhrnutim - to sa tu uz proste akosi ocakava. Zvaz aj ci tie dalsie dve seckie nedas s hviezdickou, alebo ako paragraph

MR: DONE

V tejto práci sme vykonali analýzu rôznych existujúcich ontologických riešení v oblasti publikovania spravodajských dát o bezpečnostných hrozbách. Nakoniec sme sa zamerali na ontologické riešenie UCO, ktoré sme najprv vyhodnotili bzhľadom na CTI model. Následne sme zhodnotili, aké časti nie sú dostatočne zapracované v tejto ontológii a túto časť sme rozšírili o ontológiu, ako aj o dáta z verejných databázových riešení OVAL a CVE, ktoré popisujú zraniteľnosti operačných systémov, ich platforiem a produktov.

Týmto rozšírením sme získali úplnejšiu ontológiu, ktorá vo väčšej miere pokrýva CTI model, ktorý bol naším referenčným modelom.

MH ↑ preskumal a zhodnotil si *viacero* ontlogických rieseni, nie len jedno. Nasledne si analyzoval jedno vybrane – UCO – a identfikoval si, ze v nom chyba este to a to... (CVE a OVAL, mozes napisat aj na naco su potrebne)...

MR: popisane lepsie, dopisal som UCO

Dáta v jednotlivých databázových riešeniach sme zanalyzovali, na základe týchto dát sme vytvorili ontologické riešenie, ktoré pokrývalo všetky dáta relevantné pre publikovanie a tieto vybrané dáta sme importovali a zmapovali na naše ontologické riešenia. Nad týmito dátami sme následne vytvorili rôzne analýzy a prípady použitia na popis, čo sa v nami navrhnutých ontológiách dá zistiť.

MH ↑ Tu alebo možno (aj) v prinose, nejak nam ty z toho vypadol CTI model, ktorý sme použili ako referenčný, a vyhodnotili sme UCO vzhľadom na tento model, a to rozšírenie sme tiež robili na základe toho, čo bolo ešte potrebné reprezentovať podľa CTI, nie? Tým pádom v prinose môžeme povedať, že naša ontológia je úplnejšia vzhľadom na CTI, čo je ak iste veľmi vítane... (Ale môžem byť aj ja mimo, ty do toho vidíš lepšie...)

Prínos

Pomocou ontologického riešenia, časti z domény kybernetickej bezpečnosti, sa nám podarilo vyjadriť tieto dáta vo jednotnej forme, prepojiť naše riešenie s už existujúcim štandardom reprezentovania znalostí v doméne kybernetickej bezpečnosti ako aj jej rozšírenie o časť zraniteľnosti a ich podrobný popis. Podarilo sa nám premapovať existujúce OVAL a CVE dáta na naše ontologické riešenie a poprepájať tieto dáta medzi sebou.

Vďaka tomu, vieme tieto dáta reprezentovať a publikovať na webe, v sieti prepojených dát vo forme, ktorá je prepoužiteľná a strojovo čitateľná. Hoci kto sa môže na naše dáta odkazovať, opätovne ich využiť alebo rozšíriť.

MH: ↑ Napis nie len reprezentovať, ale aj publikovať na webe, v sieti LOD... To je predsa jeden z našich cieľov

MR: pridane

Naša práca taktiež dokáže pomôcť pri prevencii a ochrane pred kybernetickými hrozbami. Vďaka dátam, ktoré sme importovali do našej

ontológie, vieme vyhodnocovať jednotlivé zraniteľnosti pre operačné systémy, platformy a produkty a taktiež vieme povedať, aké všetky produkty ovplyvňuje CVE alebo OVAL záznam. Vieme identifikovať záznamy, ktoré sa týkajú iba platformou všeobecne, alebo také, ktoré sa využívajú najčastejšie.

MH: ↑ Tato posledná časť je najviac vágna, aké „rozne“ analýzy, vieš vytvoriť viac konkrétnejšie a zakoreniť na napr. v tých prípadoch použitia?

MR: pridaný podrobnejší popis, zakomponovanie prípadov použitia

Budúce pokračovanie

MH: Kde vidíme priestor, že by s tým niekto mohol robiť, ďalšie možné vylepšenia

OVAL databáza poskytuje aj popis jednotlivých súd testov, pomocou ktorých sa vyhodnocuje, či je systém ohrozený alebo nie. Na spojenie týchto dát s našou ontológiou by bolo potrebné hlbšie preskúmať logickú sémantiku ich jazyka. V našej ontológii už sú predpripravené mapovania na tieto testy.

Možným pokračovaním je aj možnosť rozšíriť ontológiu UCO o ďalšie mapovania na už existujúce databázy, kde môžu už mať ontologické riešenie alebo takéto ontologické riešenie bude potrebné vytvoriť. Následne extrahovať relevantné dáta z takejto databázy a pokúsiť sa namapovať ich na nami nainportované dáta.

MH: ↑ Ked to takto napises, tak si nahral komissi aby sa ta spytala, ze preco si to nespracoval, a ci si to nahodou nemal uronit v ramci svojich cielov tiez. Zvaz nejaku formulaciu, ze OVAL databaza obsahuje este aj taketo data, ale na ich spojenie s nasou databazou by bolo potrebne hlbsie preskumat logicku semantiku ich „proprietaryho“ jazyka, alebo nieco podobne – z toho potom bude jasne aj co je ta praca, ktoru by bolo este potrebne v nadvaznosti urobit (a ze je to netrivialne). Uvodzovky na „proprietary“ som dal preto, lebo ten standard je zrejme verejny, ale nekoresponduje s existujucimi standardmi v oblasti ontologii

MR: zmenena prva cast k buducemu pokracovaniu

Nad nami vygenerovanými dátami je možné urobiť ďalšie analýzy, ktoré by mohli napomôcť pri prevencii pred kybernetickými útokmi.

MH: ↑ Opat, toto je velmi velmi vagne, a moze to budit dojem, ze ani nam samym to nie je uplne jasne.

MR: ina formulacia zaveru

MH: ↑ No a co mi tu uplne chyba, je spomenut sirsi kontext projektu ORBIS, vďaka ktorému na zaklade Tvojej ontologie vznikde semaniticky repozitar, do ktoreho skutocne budu tie data naimportovane a zverejnene.

Literatúra

- [BHBL11] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data: The story so far. In *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227. IGI Global, 2011.
- [BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific american*, 284(5):34–43, 2001.
- [GOS09] Nicola Guarino, Daniel Oberle, and Steffen Staab. What is an ontology? In *Handbook on ontologies*, pages 1–17. Springer, 2009.
- [IBN⁺15] Michael Iannacone, Shawn Bohn, Grant Nakamura, John Gerth, Kelly Huffer, Robert Bridges, Erik Ferragut, and John Goodall. Developing an ontology for cyber security knowledge graphs. In *Proceedings of the 10th Annual Cyber and Information Security Research Conference*, pages 1–4, 2015.
- [KM] Marja-Riitta Koivunen and Eric Miller. Semantic web – layers. <https://www.w3.org/2001/12/semweb-fin/w3csw>. Navštívené: 01.02.2020.
- [Lab17] UMBC Ebiqurity Lab. Uco2. <https://github.com/Ebiqurity/uco2>, 2017.

- [MB17] Vasileios Mavroeidis and Siri Bromander. Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence. In Joel Brynielsson, editor, *European Intelligence and Security Informatics Conference, EISIC 2017, Athens, Greece, September 11-13, 2017*, pages 91–98. IEEE Computer Society, 2017.
- [OCM12] Leo Obrst, Penny Chase, and Richard Markeloff. Developing an ontology of the cyber security domain. In *STIDS*, pages 49–56, 2012.
- [OCWM14] Alessandro Oltramari, Lorrie Faith Cranor, Robert J Walls, and Patrick D McDaniel. Building an ontology of cyber security. In *STIDS*, pages 54–61. Citeseer, 2014.
- [PUJF03] John Pinkston, Jeffrey Undercoffer, Anupam Joshi, and Timothy Finin. A target-centric ontology for intrusion detection. In *Procs. of the IJCAI-03 Workshop on Ontologies and Distributed Systems*, 2003.
- [sch] Schema.org. <https://schema.org/>. Navštívené: 27.04.2021.
- [SPF⁺16] Zareen Syed, Ankur Padia, Tim Finin, M. Lisa Mathews, and Anupam Joshi. UCO: A unified cybersecurity ontology. In David R. Martinez, William W. Streilein, Kevin M. Carter, and Arunesh Sinha, editors, *Artificial Intelligence for Cyber Security, Papers from the 2016 AAAI Workshop, Phoenix, Arizona, USA, February 12, 2016*, volume WS-16-03 of *AAAI Workshops*. AAAI Press, 2016.

- [SRa] A.T. Schreiber and Raimond. Ontotext ad, sparql overview. <https://www.w3.org/TR/2013/REC-sparql11-overview-20130321/>. Navštívené: 01.02.2020.
- [SRb] A.T. Schreiber and Raimond. Rdf framework. <https://www.w3.org/TR/rdf11-primer/>. Navštívené: 01.02.2020.
- [SW15] Malek Ben Salem and Chris Wacek. Enabling new technologies for cyber security defense with the icas cyber security ontology. In *STIDS*, pages 42–49, 2015.
- [TPKN18] Takeshi Takahashi, Bhola Panta, Youki Kadobayashi, and Koji Nakao. Web of cybersecurity: Linking, locating, and discovering structured cybersecurity information. *Int. J. Communication Systems*, 31(3), 2018.

Zoznam obrázkov

2.1	Semantic Web - vrstvy. Zdroj: [KM]	4
2.2	Príklad grafovej databázy.	8
3.1	Príklad bočného panelu vo vyhľadávači Google.	14
4.1	CTI model. Zdroj: [MB17]	23
5.1	UCO ontológia – časť Identita	33
5.2	UCO ontológia – časť Útok	35
5.3	UCO ontológia – časť Slabiny	37
6.1	CVE ontológia	40
6.2	OVAL ontológia	43