The SEPSES Knowledge Graph: An Integrated Resource for Cybersecurity

Elmar Kiesling[1(✉)], Andreas Ekelhart[1,2], Kabul Kurniawan[1,3], and Fajar Ekaputra[1,4]

[1] TU Wien, Favoritenstraße 9-11, Vienna, Austria
elmar.kiesling@tuwien.ac.at
[2] SBA Research, Favoritenstraße 16, Vienna, Austria
[3] University of Vienna, Währingerstraße 29, Vienna, Austria
[4] CDL-SQI - TU Wien, Favoritenstraße 9-11, Vienna, Austria

Abstract. This paper introduces an evolving cybersecurity knowledge graph that integrates and links critical information on real-world vulnerabilities, weaknesses and attack patterns from various publicly available sources. Cybersecurity constitutes a particularly interesting domain for the development of a domain-specific public knowledge graph, particularly due to its highly dynamic landscape characterized by time-critical, dispersed, and heterogeneous information. To build and continually maintain a knowledge graph, we provide and describe an integrated set of resources, including vocabularies derived from well-established standards in the cybersecurity domain, an ETL workflow that updates the knowledge graph as new information becomes available, and a set of services that provide integrated access through multiple interfaces. The resulting semantic resource offers comprehensive and integrated up-to-date instance information to security researchers and professionals alike. Furthermore, it can be easily linked to locally available information, as we demonstrate by means of two use cases in the context of vulnerability assessment and intrusion detection.

Keywords: Knowledge graph · Cybersecurity · Security vocabularies · Security standards · Security analysis · Intrusion detection

1 Introduction

Security and privacy have become key issues in today's modern societies characterized by a strong dependence on Information and Communication Technologies (ICT). Security incidents, such as ransomware and data theft, are widely reported in the media and illustrate the ongoing struggle to protect ICT systems. In their mission to secure systems, security professionals rely on a wealth of information such as known and newly identified vulnerabilities, weaknesses,

C. Ghidini et al. (Eds.): ISWC 2019, LNCS 11779, pp. 198–214, 2019.
https://doi.org/10.1007/978-3-030-30796-7_13

threats, and attack patterns. Such information is collected and published by, e.g., Computer Emergency Response Teams (CERTs), research institutions, government agencies, and industry experts. Whereas a lot of relevant information is still shared informally as text, initiatives to make security information available in well-defined structured formats, largely driven by MITRE[1] and NIST[2], have made significant progress and resulted in a wide range of standards [1]. These standards define high-level schemas for cybersecurity information and have resulted in various structured lists that are available for browsing on the web and for download in heterogeneous structured formats. This wealth of cybersecurity data is highly useful, but the current approach for sharing it is associated with several limitations: First, individual entities and data sets remain isolated and cannot easily be referenced and linked from other data sets. Second, whereas the governed schemas provide a well-defined structure, the semantics are not as well-defined. This limits the potential for integration and automated machine interpretation. Consequently, the resulting abundance of data raises challenges for security analysts and professionals who have to keep track of all the available sources and identify relevant information within them.

In this paper, we propose that integrating cybersecurity information into a regularly updated, public knowledge graph can overcome these limitations and open up exciting opportunities for cybersecurity research and practice. Thereby, it is possible not only to query public cybersecurity information, but also to use it to contextualize local information. As we illustrate with two example use cases in this paper, this facilitates applications such as *(i)* improved vulnerability assessment by automatically determining which new vulnerabilities affect a given infrastructure, and *(ii)* improved incident response through better contextualization of intrusion detection alerts.

Our main contributions can be summarized as follows: For cybersecurity research and practice, we advance the state of the art by providing an integrated up-to-date view on cybersecurity knowledge in a semantically explicit representation. Furthermore, we provide tools and services to query and make use of this interlinked knowledge graph. From a semantic web research perspective, we illustrate how Linked Data principles can be applied to combine local and public knowledge in a highly dynamic environment characterized by fast-changing, dispersed, and heterogeneous information. To this end, we develop an ETL pipeline that integrates newly available structured data from public sources into the knowledge graph, which involves acquisition, extraction, lifting, linking, and validation steps. We provide the following resources[3]: *(i)* vocabularies for the rich representation and interlinking of security-related information based on five well-established standards in the cybersecurity domain. *(ii)* a comprehensive SEPSES Cybersecurity Knowledge Graph (KG)[4] with detailed instance data[5]

[1] https://www.mitre.org.
[2] https://nist.gov.
[3] Available at https://w3id.org/sepses/cyber-kg.
[4] *Semantic Processing of Security Event Streams* is an ongoing research project.
[5] 36,594,388 triples as of July 2, 2019.

accessible through multiple interfaces. *(iii)* an ETL workflow published as open source that updates the knowledge graph as new information becomes available. *(iv)* a website[6] that provides documentation, status information, and pointers to the various access mechanisms provided. *(v)* a set of services to access the data, i.e., a SPARQL endpoint, a triple pattern fragments interface, a Linked Data interface, and download options for the whole data set as well as various subsets.

This semantic approach can provide a foundation for tools and services that support security analysts in applying external security knowledge and efficiently navigating dynamic security information. Ultimately, this should contribute towards improved cybersecurity knowledge sharing and increased situational awareness, both in large organizations that have dedicated security experts who are often overwhelmed by the large amount of information, and in smaller organizations that do not have the resources to invest in specialized tools and experts.

The remainder of this paper is organized as follows: Sect. 2 provides an overview of related work; Sect. 3 covers construction and maintenance of the KG, including vocabularies, data acquisition mechanisms, and updating pipelines; Sect. 4 provides an overview of the provided mechanisms to access the data in the KG and discusses its sustainability, maintenance and extensibility; Sect. 5 illustrates the usefulness of the resource by means of two example use cases; Sect. 6 concludes the paper with an outlook on future work.

2 Related Work

Various information security standards, taxonomies, vocabularies, and ontologies have been developed in academia, industry, and government agencies. In this section, we review these lines of related work, which fall into two broad categories: *(i)* standard data schemas for information sharing in the cybersecurity domain (covered in Sect. 2.1) and *(ii)* higher-level conceptualizations of security knowledge (covered in Sect. 2.2). We conclude the section by identifying the gap between those strands of work.

2.1 Standard Data Schemas

Efficient information exchange requires common standards, particularly in highly diverse and dynamic domains such as cybersecurity. Hence, a set of standards has emerged that define the syntax of description languages for structured cybersecurity information and the semantics associated with those descriptions in natural language. Some of these standards are driven by traditional standardization bodies such as ISO, ITU, IEEE or IETF. The majority, however, are contributed by open source communities or other entities such as MITRE[7], a not-for-profit research and development cooperation.[8]

[6] https://sepses.ifs.tuwien.ac.at.
[7] https://www.mitre.org.
[8] For a review of standards for the exchange of security information, cf. [1].

Salient examples for information sharing standards, all of which are integrated in the knowledge graph presented in this paper, include Common Vulnerabilities and Exposures (CVE)[9] for publicly known vulnerabilities, Common Attack Pattern Enumeration and Classification (CAPEC)[10] for known attack patterns used by adversaries, Common Weakness Enumeration (CWE)[11] for software security weaknesses, Common Platform Enumeration (CPE)[12] for encoding names of IT products and platforms, and Common Vulnerability Scoring System (CVSS)[13] for vulnerability scoring. These standards are widely used by security practitioners and integrated into security products and services, but they also serve as an important point of reference for research.

2.2 Security Ontologies

A related line of academic research aims at a high-level conceptualization of information security knowledge, which has resulted in numerous ontologies (e.g., [2,3,6,7,10,11,15]) that typically revolve around core concepts such as *asset*, *threat*, *vulnerability*, and *countermeasure*. The resulting security ontologies are typically scoped for particular application domains (e.g., risk management, incident management). The high-level ontology developed in [8], for instance, mainly focuses on malware and aspects such as *actors*, *victims*, *infrastructure*, and *capabilities*. The authors argue that expressive semantic models are crucial for complex security applications and name Open Vulnerability and Assessment Language (OVAL), CPE, Common Configuration Enumeration (CCE), and CVE as the most promising starting points for the development of a cybersecurity ontology. Inspired by that work, Oltramari et al. [9] introduce an ontological cyber security framework that comprises a top-level ontology based on DOLCE, a mid-level ontology with security concepts (e.g., *threat*, *attacker*, *vulnerability*, *countermeasure*), and a domain ontology of cyber operations including defensive and offensive actions. A comprehensive survey and classification of similar security ontologies can be found in [12].

More recently, various initiatives aimed at developing security ontologies that cover the standard schemas outlined in Sect. 2.1, including an ontology for CVE vulnerabilities [4,16,17] that can be used to identify vulnerable IT products. Ulicny et al. [14] take advantage of existing standards and markup languages such as Structured Threat Information eXpression (STIX), CAPEC, CVE and CybOX and transform their respective XML schemas through XSLT translators and custom code into a Web Ontology Language (OWL) ontology. Furthermore, they integrate external information, e.g., on persons, groups and organizations, IP addresses (WhoIs records), geographic entities (GeoNames), and "killchain" phases. In an application example, the authors illustrate how this can help to

[9] https://cve.mitre.org.
[10] https://capec.mitre.org.
[11] https://cwe.mitre.org.
[12] https://cpe.mitre.org.
[13] https://www.first.org/cvss/.

inspect intrusion detection events, e.g., by mapping events to kill chain stages and obtaining more information about threat actors based on IP addresses.

As part of a research project (STUCCO), Iannacone et al. [5] outline an approach for a cybersecurity knowledge graph and note that they aim to integrate information from both structured and unstructured data sources. Some extraction code and JSON schema data is available on the project website[14] but no integrated knowledge graph has been published. In a similar effort, Syed et al. [13] integrate heterogeneous knowledge schemas from various cybersecurity systems and standards and create a Unified Cybersecurity Ontology (UCO) that aligns CAPEC, CVE, CWE, STIX, Trusted Automated eXchange fo Indicator Information (TAXII)[15] and Att&ck[16]. Whereas most ontologies proposed in the literature are not publicly available, UCO is offered for download[17], including some example instances from industry standard repositories. However, the instance data in the dump is neither complete nor updated, and there is no public endpoint available. Finally, the Cyber Intelligence Ontology[18] is another example of an ontology that is available for download in RDF and offers classes, properties and restrictions on many industry standards, but no instance data.

Overall, a review of related work shows that although basic concepts in the cybersecurity domain have been formalized repeatedly, no model has so far emerged as a standard. Furthermore, the proposed high-level conceptualizations typically lack concrete instance information.

On the other hand, there are many standards for cybersecurity information sharing and the information is published in various structured formats[19], navigable on the web and/or available for download; however, there is no integrated view on this scattered, heterogeneous information. Hence, each application that makes use of the published data has to parse and interpret each source individually, which makes reuse, machine interpretation, and integration with local data difficult. In the following section, we describe how an evolving cybersecurity knowledge graph that provides an integrated perspective on the cybersecurity landscape can fill this gap.

3 Knowledge Graph Construction and Evolution

To construct and regularly update the SEPSES Cybersecurity KG, we define a set of vocabularies, described in Sect. 3.1, and an architecture for initial ingestion and incremental updating of the graph, covered in Sect. 3.2. Publication via Linked Data (LD), Triple Pattern Fragments (TPF), a SPARQL endpoint, and RDF dumps are covered in Sect. 4.

[14] https://github.com/stucco.
[15] https://oasis-open.github.io/cti-documentation/.
[16] https://attack.mitre.org.
[17] https://github.com/Ebiquity/Unified-Cybersecurity-Ontology.
[18] https://github.com/daedafusion/cyber-ontology.
[19] Most commonly as XML or JSON files.

3.1 Conceptualization and Vocabularies

To model the domain of interest, we started with a survey and found that the vast majority of conceptualizations described in the literature are not available online. Those that were available did not provide sufficiently detailed classes and properties to represent all the information available in the cybersecurity repositories we target.

Hence, we opted for a bottom-up approach starting from a set of well-established industry data sources. We structured our vocabularies based on the schemas used to publish existing instance data and chose appropriate terms based on the survey of existing conceptualizations. In choosing this approach, our main design goal was to include the complete information from the original data sources and make the resulting knowledge graph self-contained. To facilitate mapping to other existing conceptualizations, we kept the Resource Description Framework (RDF) model structurally similar to the data models of the original sources. This should make it easy for users already familiar with the original data sources to navigate and integrate our semantic resource. Furthermore, we can easily refer to the original documentation and examples in the vocabularies. We then created a schema that covers the following security information repositories (cf. Fig. 1 for a high-level overview).[20]

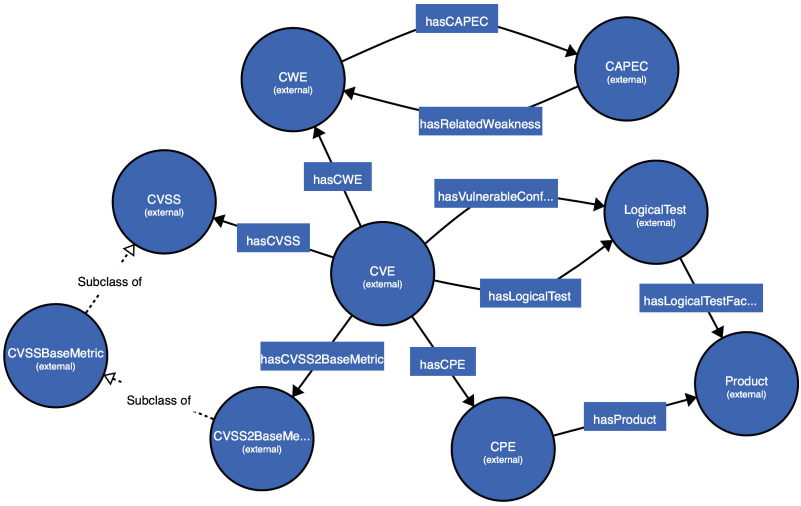

Fig. 1. SEPSES knowledge graph vocabulary high-level overview

CVE is a well-established industry standard that provides a list of identifiers for publicly known cybersecurity vulnerabilities. In addition to CVE, we

[20] The figure omits detailed concepts for the sake of clarity. The complete vocabularies can be found at https://github.com/sepses/vocab.

integrate the National U.S. Vulnerability Database (NVD), which enriches CVEs with additional information, such as security checklist references, security-related software flaws, misconfigurations, product names, and impact metrics. We represent this information in the CVE class, which includes data type properties such as `cve:cveId`, `cve:description`, `cve:issued` and `cve:modified` timestamps. Based on the NVD information, we can link CVE to affected products (`cve:hasCPE`), vulnerable configurations (`cve:hasVulnerableConfiguration`), impact scores (`cve:hasCVSS`), related weaknesses (`cve:hasCWE`), and external references (`cve:hasReference`).

CVSS provides a quantitative model to describe characteristics and impacts of IT vulnerabilities. It is well-established as a standard measurement system for organizations worldwide. We integrate the CVSS scores provided by NVD, and model the CVSS metrics by means of the CVSSBaseMetric, CVSSTemporalMetric, and CVSSEnvironmentalMetric classes to comply with the CVSS specification[21].

CPE provides a structured naming scheme for IT systems, software, and packages based on URIs. NIST hosts and maintains the CPE Dictionary, which currently is based on the CPE 2.3 specification. We represent CPEs with the CPE class and reference product information with `cpe:hasProduct`. Furthermore, we define a set of properties that describe a product, such as product name, version, update, edition, language, etc. The vendor of each product is modeled as a Vendor and referenced by `cpe:hasVendor`.

CWE is a community-developed list of common software security weaknesses that contains information on identification, mitigation, and prevention. NVD vulnerabilities are mapped to CWEs to offer general vulnerability information. This information is modeled using the CWE class and a set of datatype properties such as `cwe:id`, `cwe:name`, `cwe:description`, and `cwe:status`, as well as object properties, to e.g., link applicable platforms (`cwe:hasApplicablePlatform`), attack patterns (`cwe:hasCAPEC`), consequences (`cwe:hasCommonConsequence`), related weaknesses to model the CWE hierarchy (`cwe:hasRelatedWeakness`) and potential mitigations (`cwe:hasPotentialMitigation`).

CAPEC is a dictionary of known attack patterns used by adversaries to exploit known vulnerabilities, and can be used by analysts, developers, testers, and educators to advance community understanding and enhance defenses. We model CAPEC patterns in the CAPEC class with datatype properties such as `capec:id`, `capec:name`, `capec:likelihoodOfAttack`, and `capec:description`. Additional information is linked via object properties such as consequences `capec:hasConsequences`, required skills `capec:hasSkillRequired`, attack prerequisites `capec:prerequisites`, and attack consequences `capec:hasConsequence`.

Most of these data sets define identifiers for key entities such as vulnerabilities, weaknesses, and attack patterns and reuse some concepts from other standards (e.g., CPE names and CVSS scores are used within CVE). In the next section, we will describe how we leverage these references to link the data.

[21] https://www.first.org/cvss/specification-document.

3.2 ETL Process

Figure 2 illustrates the overall architecture and the data acquisition, resource extraction, entity linking and validation, storage and publication steps necessary to provide a continuously updated cybersecurity knowledge graph. In the following, we describe the steps in the core Extraction, Transformation, Loading (ETL) process that periodically checks and digest data from the various sources.

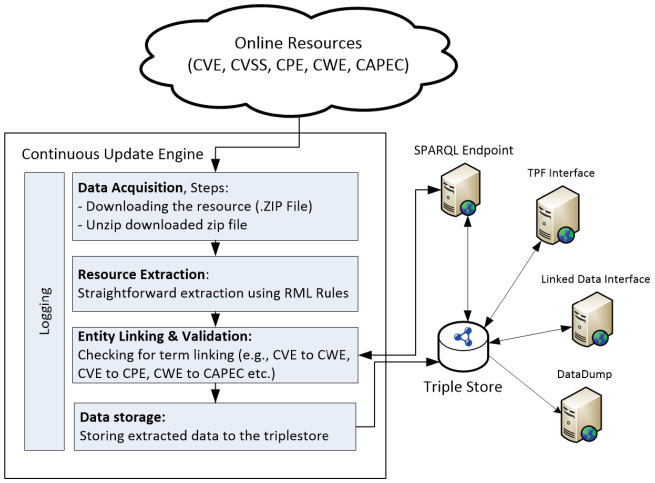

Fig. 2. Architecture: ETL process and publishing

Data Acquisition. We populate our KG using data from various sources that provide data on their respective web sites for download in heterogeneous formats such as CSV, XML, and JSON. These cybersecurity data sources are updated regularly to reflect changes in the real-world. CVE data, for instance, is typically updated once every two hours.[22] In order to capture changes and reflect them in the knowledge graph, our ETL engine will regularly poll for updates and ingest the latest version of the sources.

Resource Extraction. We use the caRML engine[23] to transform the original source files from their various formats. Furthermore, we use Apache Jena[24] to transform the raw RDF data obtained from the RML mappings into the structure of the final ontology. Initially, we developed RDF Mapping Language (RML) transformation mappings that utilized specific features from caRML, such as *carml:multiJoinCondition*. Due address performance issues, however, we decided

[22] cf. https://nvd.nist.gov/vuln/data-feeds.
[23] https://github.com/carml/carml.
[24] https://jena.apache.org.

to restructure the initial mapping into generic RML mappings that do not involve specific constructs from caRML, which improved performance considerably.[25] Because the original data sources have an established ID system, instance ID generation was straightforward for most sources (i.e., CWE, CVE CAPEC, and CVSS). For CPE, however, the instance name is a composite of several naming elements (e.g., product name, part, vendor, version, etc.), separated by special characters. To solve the issue, we use `XPath` functions to clean and produce a unique name for each CPE instance.

Entity Linking and Validation. In this part of the ETL process, we link data from different sources based on common identifiers in the data. Each CWE weakness, for example, typically references several CAPEC attack patterns. Based on these identifiers, we create direct links between associated resources. Specifically for CPE, we deploy the exact same `XPath` functions for its identifier in the two sources (CPE and CVE) where CPE instances are generated, to make sure that these data can be linked correctly. To ensure data quality, we validate the generated RDF with SHACL to make sure that the necessary properties are included for each generated individual. Furthermore, we validate whether the resulting resources are linked correctly, as references to identifiers that are not or no longer available in other data sets are unfortunately a common issue. As an example, a CVE instance may have a relation to another resource such as a CPE identifier. In this case, the validation mechanism will check whether the referenced CPE instance exists in the extracted CPE data, log missing instances and create temporary resources for them.

Data Storage. We store the extracted data in a triple store and generate statistics such as parsing time, parsing status (success or fail), counts of instances, links, and generation time. To make sure that the data is continuously up to date, we wrote a set of bash scripts that are set to be executed in regular intervals to trigger the knowledge generation process and store the result in the triple store. To date, this resulted in more than half a million instances and 36 million triples; Table 1 provides a breakdown of the generated data.

Table 1. SEPSES knowledge graph statistics (As per July 2, 2019.)

	CVE	CVSS	CPE	CWE	CAPEC	SnortRules
Axioms	68	248	111	256	149	486
Class count	7	9	5	10	8	10
Object property Count	6	8	4	9	6	10
Data property count	8	37	18	40	22	103
Individual count	123,005	123,220	393,695	808	516	3,488

[25] In some cases, this reduced processing time from appr. an hour to less than a minute.

4 Knowledge Graph Access

The SEPSES web site[26] provides pointers to the various resources covered in this paper, i.e., the LD resources[27], the SPARQL[28] and TPF query interfaces[29], a download link for the complete RDF snapshots[30], and the ETL engine source code[31]. This allows users to choose the most appropriate access mechanism for their application context.

4.1 Sustainability, Maintenance and Extensibility

The SEPSES KG is being developed jointly by TU Wien and SBA Research, a well-established research center for information security that is embedded within a network of more than 70 companies as well as 15 Universities and research institutions. Endpoints and data sets are hosted at TU Wien and maintained as part of the research project SEPSES, which aims to leverage semantic web technologies for security log interpretation. During this project, we will extend the KG and leverage it as background knowledge in research on semantic monitoring and forensic analysis.

To keep the KG in sync with the evolving cybersecurity landscape, we will continue to automatically poll and process updates of the original raw data sources. We choose our polling strategy according to the varying update intervals of the data sources: CVEs are typically updated once every two hours, CPEs are typically updated daily. CWE and CAPEC are less dynamic and are updated approximately on a yearly schedule.

Furthermore, SBA Research has an active interest in developing and diffusing the KG internally and within its partner network, which will secure long-term maintenance beyond the current research project. We also expect the KG to grow and establish an active external user community during that time. To this end, we publish our vocabularies and the source code under an open source MIT license[32] and encourage community contributions.[33] Adoption success will be measured *(i)* based on access statistics (web page access, SPARQL queries, downloads, etc.), and *(ii)* the emergence of a community around the knowledge graph (code contributions, citations, attractiveness as a linked data target, number of research and community projects that make use of it, etc.).

[26] https://w3id.org/sepses.
[27] e.g., https://w3id.org/sepses/resource/cve/CVE-2014-0160.
[28] https://w3id.org/sepses/sparql.
[29] https://ldf-server.sepses.ifs.tuwien.ac.at.
[30] https://w3id.org/sepses/dumps/.
[31] https://github.com/sepses/cyber-kg-converter.
[32] https://opensource.org/licenses/MIT.
[33] The original raw data are published by MITRE with a no-charge copyright license and by NVD without copyright.

5 Use Cases

In this section, we illustrate the applicability of the cybersecurity knowledge graph by means of two example scenarios.

5.1 Vulnerability Assessment

In security management, identifying, quantifying, and prioritizing vulnerabilities in a system is a key activity and a necessary precondition for threat mitigation and elimination and hence for the successful protection of valuable resources. This Vulnerability Assessment (VA) process can involve both active techniques such as scanning and penetration testing and passive techniques such as monitoring the wealth of public data sources for relevant vulnerabilities and threats. For the latter, keeping track of all the relevant information and determining relevance and implications for the assets in a system is a challenging task for security professionals. In this scenario, we illustrate how the developed knowledge graph can support security analysts by linking organization-specific asset information to a continuously updated stream of known vulnerabilities.

Setting: To illustrate the approach, we modeled a simplified example network comprising of three HOSTs – two workstations, a server – and NETWORKDEVICEs. All hardware components are sub classes of ITASSETs. Furthermore, we model the software installed on each HOST by means of the `hasInstalledProduct` property that links the host to a CPE specification. To determine the potential severity of an impact, we also include DATAASSETs, their `classification` (*public*, *private*, *restricted*), and their storage location (`storedOn` HOST) in the model. In practice, the modeling of a system can be supported by existing IT asset/software discovery and inventory tools.

Query 1: Once a model of the local system has been created, the vulnerability information published in the cybersecurity knowledge graph can be applied and contextualized by means of a federated SPARQL query. Note that we also provide a TPF interface for efficient querying. In particular, a security analyst may be interested in all known vulnerabilities that potentially apply to each host, based on the software that is installed on it (cf. Listing 1). Table 2 shows an example query result. Each resource in the table points to its Linked Data representation, which can serve as a starting point for further exploration. Note that as new vulnerability information becomes available and is automatically integrated into the knowledge graph through the process described in Sect. 3, the query results will automatically reflect newly identified vulnerabilities.

Table 2. Vulnerability assessment query 1 – results

hostName	IP	product	cveIds
DBServer1	192.168.1.3	Windows Server 2016	CVE-2016-3332, ..., CVE-2017-8746
Workstation1	192.168.1.1	Windows 10	CVE-2016-3302, ..., CVE-2015-2554

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX asset: <http://w3id.org/sepses/vocab/bgk/assetKnowledge#>
PREFIX cve: <http://w3id.org/sepses/vocab/ref/cve#>
PREFIX cpe: <http://w3id.org/sepses/vocab/ref/cpe#>
PREFIX cvss: <http://w3id.org/sepses/vocab/ref/cvss#>
PREFIX cwe: <http://w3id.org/sepses/vocab/ref/cwe#>

SELECT  distinct ?hostName str(?ip) as ?IP ?product
(group_concat(?cveId) as ?cveIds)  from
<http://localhost:8890/localdata2>
WHERE {
      ?s a asset:Host.
      ?s rdfs:label ?hostName.
      ?s asset:ipAddress ?ip.
      ?s asset:hasProduct ?p.
SERVICE <http://sepses.ifs.tuwien.ac.at/sparql> {
      ?cve cve:hasCPE ?p .
      ?cve cve:id ?cveId.
          ?p cpe:title ?product .
  }
}
group by ?hostName ?ip ?product
```

Listing 1: Vulnerability Asessment Query 1 – Vulnerable Assets

Query 2: In order to assess the potential impact that a newly identified vulnerability may have, it is critical to asses which data assets might be exposed if an attacker can successfully exploit it. In the next step, we hence take advantage of the modeled data assets and formulate a query (cf. Listing 2)[34] to retrieve the most severe vulnerabilities, i.e., those that affect hosts that store *sensitive private* data (classification value = 1) and have a *complete* confidentiality impact (as specified in CVSS). Table 3 shows the query result and illustrates how such immediate analysis can save time by avoiding manual investigation steps.

Exploration: The query results can serve as a starting point for further exploration of the Linked Data in the knowledge graph[35]. By navigating it, a security analyst can access information from various sources such as, e.g., attack prerequisites and potential mitigations from CAPEC, weakness classifications and potential mitigations from CWE, and scorings from CVSS.

Table 3. Vulnerability assessment query 2 – results

hostName	cveId	conf	score	dataAsset	class	consequence
Workstation2	2016-1646	COMPLETE	9.3	EmpData	Private	Read Memory
Workstation2	2016-1653	COMPLETE	9.3	EmpData	Private	DoS: Crash, Exit...
Workstation2	2016-1583	COMPLETE	7.2	EmpData	Private	DoS: Resource Cons...
Workstation2	2016-1583	COMPLETE	9.3	EmpData	Private	Execute Unauthorized ...

[34] Prefixes identical to Listing 1.
[35] e.g., https://w3id.org/sepses/resource/cve/CVE-2016-1646.

```
SELECT  DISTINCT ?hostName  ?cveId
?confidentiality as ?conf ?cvssScore AS ?score ?dataAsset ?classification AS ?class
↪  ?consequence
FROM <http://localhost:8890/localdata>
WHERE {
    ?s a asset:Host.
    ?s rdfs:label ?hostName.
    ?s asset:hasProduct ?product.
    ?s asset:hasDataAsset ?dt.
    ?dt rdfs:label ?dataAsset.
    ?dt asset:hasClassification ?c.
    ?c rdfs:label ?classification.
    ?c asset:dataClassificationValue ?cv
FILTER (?confidentiality = "COMPLETE")
FILTER (?cv = 1)

SERVICE <http://sepses.ifs.tuwien.ac.at/sparql> {
    ?cve cve:hasCPE ?product .
    ?cve cve:id ?cveId.
    ?cve cve:hasCVSS2BaseMetric ?cvss2.
    ?cvss2 cvss:confidentialityImpact ?confidentiality.
    ?cvss2 cvss:baseScore ?cvssScore.
    ?cve cwe:hasCWE ?cwe.
    ?cwe cwe:hasCommonConsequence ?cc.
    ?cc cwe:consequenceImpact ?consequence
   }
}
```

Listing 2: Vulnerability Assessment Query 2 – Critical Vulnerabilities

5.2 Intrusion Detection

In this scenario, we illustrate how alerts from the Network Intrusion Detection System (NIDS) Snort[36] can be connected to the SEPSES Cybersecurity KG in order to obtain a deeper understanding of potential threats and ongoing attacks. As a first step, we acquired the Snort community rule set[37] and integrated it into our cybersecurity repository using a defined vocabulary[38]. Snort can monitor these rules and trigger alerts once it finds matches to these patterns in the network traffic. We represent SnortRules as a class with two linked concepts SnortRuleHeader and SnortRuleOption. For SnortRuleOption we include properties such as `sr:hasClassType` and `sr:hasCVEReference`, which will be used to link incoming alerts to CVEs.

Setting: We use a large data set collected during the MACCDC 2012[39] cybersecurity competition as a realistic set of real-world intrusion detection alerts (cf. Listing 3 for an example). We provide and use a Snort alert log vocabulary[40] to map those alerts into RDF.

[36] https://www.snort.org.
[37] https://www.snort.org/downloads.
[38] https://w3id.org/sepses/vocab/rule/snort.
[39] https://maccdc.org/2012-agenda/, source: https://www.secrepo.com.
[40] https://w3id.org/sepses/vocab/log/snort-alert.

```
[**] [1:1807:12] WEB-MISC Chunked-Encoding transfer attempt [**]
[Classification: Web Application Attack] [Priority: 1]
11/10-11:10:12.321349 10.2.189.248:54208 -> 154.241.88.201:80
TCP TTL:61 TOS:0x0 ID:36462 IpLen:20 DgmLen:1200 DF
***A**** Seq: 0xCFAD1EE0  Ack: 0xB27D1032  Win: 0xB7  TcpLen: 32
TCP Options (3) => NOP NOP TS: 2592976 143157138
```

Listing 3: IDS Alert Example from MACCDC

Query: When a Snort alert is triggered, a security expert typically has to analyze its relevance and decide about potential mitigations. False positives are common in this context. For instance, a particular attack pattern may be detected frequently in a network, but it may not be relevant if the targeted host configuration is not vulnerable. To support security analysts in this time-critical and information-intensive analysis task, we identify the corresponding Snort rule that triggered each particular alert. These rules often include a reference to a CVE, which we can use to query our knowledge graph for detailed CVE information related to an alert. Furthermore, by matching the installed software on the host to the vulnerable product configuration defined in CVE (cf. Scenario 1), we can automatically provide security decision makers a better foundation to estimate the relevance of a Snort alert wrt. to their protected assets. To illustrate this process, Listing 4[41] shows an example query to obtain CVE Ids and vulnerable products from Snort alerts. Based on the result Table 4, a security analyst can query if the attacked host has the vulnerable software installed (similar to Listing 1).

```
PREFIX cve:         <http://w3id.org/sepses/vocab/ref/cve#>
PREFIX cpe:         <http://w3id.org/sepses/vocab/ref/cpe#>
PREFIX snort:       <http://w3id.org/sepses/vocab/ref/snort#>
PREFIX snort-rule:  <http://w3id.org/sepses/vocab/rule/snort#>
PREFIX snort-alert: <http://w3id.org/sepses/vocab/log/snort-alert#>

SELECT DISTINCT ?alert ?message ?sid ?sourceIp ?destinationIp ?cveId ?cpeId
FROM <http://localhost:8890/snortalert>
WHERE {
    ?alert a snort-alert:IDSSnortAlertLogEntry ;
           snort:signatureId ?sid ;
           snort:message ?message ;
           snort:sourceIp ?sourceIp ;
           snort:destinationIp ?destinationIp .

    SERVICE <http://w3id.org/sepses/sparql> {
        ?rule a snort-rule:SnortRule ;
              snort-rule:hasRuleOption ?ruleOption .
        ?ruleOption snort:signatureId ?sid ;
                    snort-rule:hasCveReference ?cve .
        ?cve cve:id ?cveId ;
             cve:hasCPE/cpe:id ?cpeId
    }
}
```

Listing 4: Intrusion Detection query

[41] Prefixes from Listing 1 are reused.

Table 4. Intrusion detection query results

alert	message	sid	sourceIP	targetIP	cveId	cpeId
Alert001	WEB-MISC Chunked...	1807	10.2.190.254	154.241.88.201	2002-0392	cpe:/a:apa...
Alert002	WEB-MISC WebDAV...	1070	10.2.190.254	154.241.88.201	2000-0951	cpe:/a:micr...
Alert003	WEB-MISC TRACE...	2056	10.2.197.241	154.241.88.201	2004-2320	cpe:/a:bea:w...
Alert004	WEB-FRONTPAGE...	1248	10.2.190.254	154.241.88.201	2001-0341	cpe:/o:micr...
Alert005	WEB-MISC Netscape...	1048	10.2.197.241	154.241.88.201	2001-0250	cpe:/a:netsc...

6 Conclusions

In this resource paper, we highlight the need for semantically explicit representations of security knowledge and the current lack of interlinked instance data. To tackle this challenge, we present a cybersecurity knowledge graph that integrates a set of widely adopted, heterogeneous cybersecurity data sources.

To maintain the knowledge graph and integrate newly available information, we developed an ETL process that updates it as new security information becomes available. In order to make this resource publicly available and easy to use, we offer multiple services to access the data, including a SPARQL endpoint, a triple pattern fragments interface, a Linked Data interface, and download options for the complete data set.

We demonstrated the usefulness of the graph by means of two example use cases in vulnerability assessment and semantic interpretation of alerts generated by intrusion detection systems. Given the compelling need for efficient exchange of machine-interpretable cybersecurity knowledge, we expect the KG to be useful for practitioners and researchers, and hope that the resource will ultimately facilitate novel and innovative semantic security tools and services. Future work will focus on disseminating the resource in the security domain, building a community of users and contributors around it, and growing the knowledge graph by integrating additional security standards and information extracted from structured and unstructured sources.

Acknowledgments. This work has been supported by netidee SCIENCE, the Austrian Science Fund (FWF) under grant P30437-N31, and the Christian Doppler Research Association. The competence center SBA Research (SBA-K1) is funded within the framework of COMET—Competence Centers for Excellent Technologies by BMVIT, BMDW, and the federal state of Vienna, managed by the FFG.

References

1. Dandurand, et al.: Standards and tools for exchange and processing of actionable information. European Union Agency for Network and Information Security, Luxembourg (2015)
2. Ekelhart, A., Fenz, S., Neubauer, T.: Aurum: a framework for information security risk management. In: Proceedings of the 42nd Hawaii International Conference on System Sciences (2009). https://doi.org/10.1109/HICSS.2009.82

3. Fenz, S., Ekelhart, A.: Formalizing information security knowledge. In: Proceedings of the 4th International Symposium on Information, Computer, and Communications Security (2009). https://doi.org/10.1145/1533057.1533084
4. Guo, M., Wang, J.: An ontology-based approach to model common vulnerabilities and exposures in information security. In: ASEE Southeastern Section Annual Conference (2009)
5. Iannacone, M., et al.: Developing an ontology for cyber security knowledge graphs (2015). https://doi.org/10.1145/2746266.2746278
6. Kim, A., Luo, J., Kang, M.: Security ontology for annotating resources. In: Meersman, R., Tari, Z. (eds.) OTM 2005. LNCS, vol. 3761, pp. 1483–1499. Springer, Heidelberg (2005). https://doi.org/10.1007/11575801_34
7. Martimiano, A., Moreira, E.S.: An owl-based security incident ontology. In: Proceedings of the Eighth International Protege Conference (2005)
8. Obrst, L., Chase, P., Markeloff, R.: Developing an ontology of the cyber security domain. In: Proceedings of the 7th International Conference on Semantic Technologies for Intelligence, Defense, and Security (2012)
9. Oltramari, A., Cranor, L., Walls, R., McDaniel, P.: Building an ontology of cyber security. In: Proceedings of the 9th Conference on Semantic Technology for Intelligence, Defense, and Security (2014)
10. Raskin, V., Hempelmann, C., Triezenberg, K., Nirenburg, S.: Ontology in information security: a useful theoretical foundation and methodological tool. In: Proceedings of the 2001 Workshop on New Security Paradigms (2001). https://doi.org/10.1145/508171.508180
11. Schumacher, M.: Toward a security core ontology. Security Engineering with Patterns. LNCS, vol. 2754, pp. 87–96. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45180-8_6
12. Souag, A., Salinesi, C., Comyn-Wattiau, I.: Ontologies for security requirements: a literature survey and classification. In: Bajec, M., Eder, J. (eds.) CAiSE 2012. LNBIP, vol. 112, pp. 61–69. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31069-0_5
13. Syed, Z., Padia, A., Mathews, M., Finin, T., Joshi, A.: UCO: a unified cybersecurity ontology. In: Proceedings of the AAAI Workshop on Artificial Intelligence for Cyber Security (2016)
14. Ulicny, B., Moskal, J., Kokar, M., Abe, K., Smith, J.: Inference and ontologies (2014). https://doi.org/10.1007/978-3-319-11391-3_9
15. Undercoffer, J., Joshi, A., Pinkston, J.: Modeling computer attacks: an ontology for intrusion detection. In: Vigna, G., Kruegel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 113–135. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45248-5_7
16. Wang, J., Guo, M.: Security data mining in an ontology for vulnerability management. In: 2009 International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing (2009). https://doi.org/10.1109/IJCBS.2009.13
17. Wang, J., Guo, M., Camargo, J.: An ontological approach to computer system security. Inf. Secur. J.: A Global Perspect. **19**(2) (2010). https://doi.org/10.1080/19393550903404902