Beyond NVD: Cybersecurity meets the Semantic Web.

Raúl Aranovich
University of California, Davis
Davis, USA
raranovich@ucdavis.edu

Muting Wu
University of California, Davis
Davis, USA
mtiwu@ucdavis.edu

Dian Yu
University of California, Davis
Davis, USA
diayu@ucdavis.edu

Katya Katsy
University of California, Davis
Davis, USA
kkatsy@ucdavis.edu

Benyamin Ahmadnia
Harvard University
Cambridge, USA
bahmadniayebosari@bwh.harvard.edu

Matthew Bishop
University of California, Davis
Davis, USA
mabishop@ucdavis.edu

Vladimir Filkov
University of California, Davis
Davis, USA
vfilkov@ucdavis.edu

Kenji Sagae
University of California, Davis
Davis, USA
sagae@ucdavis.edu

ABSTRACT

Cybersecurity experts rely on the knowledge stored in databases like the NVD to do their work, but these are not the only sources of information about threats and vulnerabilities. Much of that information flows through social media channels. In this paper we argue that security experts and general users alike can benefit from the technologies of the Semantic Web, merging heterogeneous sources of knowledge in an ontological representation. We present a system that has an ontology of vulnerabilities at its core, but that is enhanced with NLP tools to identify cybersecurity-related information in social media and to launch queries over heterogeneous data sources. The transformative power of Semantic Web technologies for cybersecurity, which has been proven in the biomedical field, is evaluated and discussed.

CCS CONCEPTS

• **Security and privacy → Vulnerability management**; • **Information systems**;

KEYWORDS

cybersecurity, ontology, nlp, neural networks, social media

ACM Reference Format:
Raúl Aranovich, Muting Wu, Dian Yu, Katya Katsy, Benyamin Ahmadnia, Matthew Bishop, Vladimir Filkov, and Kenji Sagae. 2021. Beyond NVD: Cybersecurity meets the Semantic Web.. In *New Security Paradigms Workshop (NSPW '21), October 25–28, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3498891.3501259

NSPW '21, October 25–28, 2021, Virtual Event, USA
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8573-2/21/10.
https://doi.org/10.1145/3498891.3501259

1 INTRODUCTION

The National Vulnerability Database (NVD) was created in 2005 as a comprehensive database of vulnerabilities, with funding from the Department of Homeland Security. It was a continuation of a project that started in 1999, the Internet Categorization of Attack Toolkit (I-CAT).[1] The NVD is a set of "databases of security checklist references, security related software flaws, misconfigurations, product names, and impact metrics[2]". The NVD does not perform security checks. Rather, once a vulnerability is discovered and assigned a unique identifier in the Common Vulnerabilities and Exposures (CVE) dictionary, the NVD analyzes it to assign it a class, a severity score, and to specify the systems it may affect.

The NVD database has had a positive impact on the cybersecurity research community, but in the two decades that have elapsed since the effort began several limitations in its scope, methodology, and functionality have come to light. Part of the problem lies in the sheer volume of vulnerabilities that are discovered and reported, which overwhelms the resources available for their analysis, classification, and dissemination [4]. Moreover, the NVD is eclipsed and outrun by the torrent of information about vulnerabilities that now runs through Twitter, Stack Overflow, Reddit, and other social media channels (including the old-fashioned web page posting a security advisory).

We propose a new approach to the public disclosure of vulnerabilities that removes the limitations of the NVD, while at the same time building on its strengths. Several new developments in the field of cybersecurity and related technologies have matured to the point that we can move beyond the NVD by bringing the full force of the Semantic Web (enhanced with NLP methods) to it. Following Hitzler [14], we conceive of the Semantic Web not as an artifact (i.e. a collection of hyperlinked WWW pages with semantic markup), but as an ensemble of technologies that allow for information to be shared, searched, and reasoned upon over the internet. This approach will give vulnerability managers and other members of

[1] I-CAT was initially an access database of attack scripts. Attack scripts are necessary to probe the safety of a system, but they were as difficult to obtain as they were ubiquitous. The purpose of I-CAT was to facilitate the acquisition of attack scripts on behalf of security professionals so that they could conduct their trade [20].
[2] https://nvd.nist.gov/general

incident response teams better and faster access to the information they need to perform their jobs. For instance, easier sharing and search can, e.g., benefit faster adoption of patches for end users who depend on social platforms, while integrated reasoning can enable non-obvious conclusions (e.g. regarding dependencies) through a dialog with end users on the need and ease of patching.

To be explicit, our contribution is the use of the Semantic Web to augment existing data on vulnerabilities predictions, identification, an remediations. We use linguistic tools to augment the technical information extant in vulnerabilities databases such as the NVD and CVE. This also simplifies the tasks of analysts by bringing together information from various sources into one easily digested system.

At the core of any Semantic Web-based system is an ontology: a formal (that is, explicit, consistent, and complete) representation of the accumulated knowledge in a particular field in the form of entities and classes with their relations, and that allows for implicit knowledge to be extracted by reasoning. Our ontology uses the structured information in the NVD as a starting point, expanding its coverage in different directions. We will argue that a well-crafted ontology can incorporate many of the improvements to the NVD that have been proposed in the recent literature. As a proof of concept, we introduce TRONTO, a system built around an ontology of cybersecurity, which retrieves vulnerability-related information from social media and supports queries.

The paper is structured as follows: Section 2 reviews recent literature on the strengths and weaknesses of the NVD, using it to frame our proposal. Section 3 discusses the use of ontologies in the Semantic web, the development of ontologies for cybersecurity, and our method of lifting the static information in the NVD to create the ontology at the core of TRONTO. Section 4 discusses three ways in which TRONTO extends the information in the NVD: I) Inferring new classes from the information already present in the database by means of a reasoner, II) enriching the relations among the concepts in the NVD, in particular to take care of the problem of the insecure dependencies [24], and III) expanding the conceptual coverage from vulnerabilities and configurations to threats, countermeasures, and other concepts and relations based on a risk model of cybersecurity [13, 32]. Section 5 introduces the methods we follow to link the instances on the ontology (which are individual CVE entries) to tweets and posts in social media, the procedures we follow to filter out and classify the stream of social media posts about cybersecurity, and the methodologies for querying the results of the searches. Even though we are currently employing text-based methods for these tasks, the potential for semantic-based methods in queries is apparent, and they lead to future work as well. In section 6 we discuss the results and lay the groundwork for future research.

2 LITERATURE REVIEW AND FRAMING OF OUR PROPOSAL

The NVD is the largest public repository of information about known software vulnerabilities. The cybersecurity research community has used the information it contains to examine the factors that contribute to the severity of a vulnerability [11, 15], or to try to predict when new vulnerabilities will emerge [36, 37]. And while security experts (a.k.a. "ethical hackers") use the information in the NVD to test the security of systems, the NVD does not seem to be a useful tool to train developers and engineers to craft more secure code [23].

The NVD has an important role to play in Vulnerability Management, one of the service areas that a Computer Security Incident Response Team (CSIRT) may attend to [10]. One of the possible functions of a CSIRT is the discovery, analysis, disclosure, and management of vulnerabilities. The CSIRT identifies vulnerabilities when responding to an incident, from public sources such as vendor announcements and social media, or from its own research. A CSIRT may also be required to disclose a vulnerability as part of its management services. The NVD provides a template for such disclosures, which may include "information such as an overview or description, a unique vulnerability identifier, impact, severity, or CVSS score, resolution (remediation or mitigation), and supporting references or materials."

Once a new vulnerability is discovered, it is given a unique identifier in the form of a CVE designation. Trained analysts enrich the CVE entry with metadata about the vulnerability's classification, potential targets, severity score, etc. All of this information is stored in the NVD. But the NVD is a victim of its own success. Having farmed out the discovery of vulnerabilities to a community of cyber-sleuths, it cannot keep up with the volume of vulnerabilities that are discovered and reported [4]. One proposed solution is to let the same agencies that first identify vulnerabilities also contribute some of the associated metadata.

The structured information contained in the NVD is also an important component of current efforts to increase transparency in the production and commercialization of software. A Software Bill of Materials (SBOM) would require developers to keep a record of the supply chain that provides upstream products (e.g. libraries) packaged with their systems. "An SBOM is effectively a nested inventory, a list of ingredients that make up software components. An SBOM identifies and lists software components, information about those components, and supply chain relationships between them." [22]. An SBOM should provide users with reliable knowledge about the safety of the systems they use, and for that it depends on external sources, like the NVD. Currently, the NVD provides information about vulnerabilities and the products they affect, in the form of a Common Platform Enumeration (CPE) dictionary. By superimposing this information on a complete supply chain tree, an SBOM would provide "a means by which to convey the transitive exploitability or exposure of a vulnerability along supply chains." But this approach highlights another important limitation of the NVD in its current form, which is that it relies on external devices to solve the problem of insecure dependencies.

A recurring theme in the literature evaluating the strengths of the NVD is whether or not it provides metrics that are useful for the prioritization of vulnerabilities in the management process. Because the cost of patching or mitigating all vulnerabilities in an enterprise is prohibitive and inefficient, security teams must have a way to decide which vulnerabilities to address first, and what kind of action to take. The NVD provides a CVSS score, which can be taken as an index for prioritization, but this approach has been criticized as highly ineffective [17, 30]. One approach, the Stakeholder-Specific Vulnerability Categorization (SSVC), replaces the severity scoring in CVSS with a qualitative evaluation of factors that weigh into prioritization for vulnerability management (e.g. attack surface, exploitability, etc.), and organizes these factors into a decision tree to arrive at a conclusion as to which vulnerabilities to address first. In a related study, Jacobs et al. compare remediation strategies based on CVSS scores and on the existence of an exploit "in the wild". They show that a balanced CVSS-based strategy still recommends patching 31K unexploitable vulnerabilities, out of 35K patched. On the other hand, an exploit-based strategy ends up recommending significantly fewer vulnerabilities for patching (about 7.9K).

The NVD continues to invite researchers to probe for inconsistencies and gaps in its content [1, 18], to evaluate the effects of these inconsistencies, and to provide solutions. But, to sum up, any comprehensive approach to revamp the NVD must fulfill the following goals: a) Reduce the involvement of analysts in extracting metadata from the documents that describe the vulnerabilities, b) Expand the information around each vulnerability and the systems they affect to make it easier for the response teams to do their job, in particular discovery and reporting, and patching and mitigation, c) Integrate the NVD with other sources of information that are critical to achieve security objectives, such as supply-chain structure and existence of exploits, d) Replace unreliable elements of the NVD (such as the CVSS), which may be misleading, with better measures and methods. We suggest that replacing the database format currently adopted by the NVD with a Semantic Web framework can go a long way to achieving those goals.

The Semantic web offers a dynamic way to organize information from heterogeneous sources in a way that overcomes the limitations inherent in text-based approaches. To achieve this goal, knowledge is represented in an ontology or knowledge graph. An ontology represents knowledge as a taxonomy of classes and instances, connected by properties and relations. There are crucial features of ontologies that solve some of the shortcomings of the NVD, since ontologies have known advantages over databases. First, ontologies are designed to integrate with each other. An ontology that relates vulnerabilities to software components could easily import an ontology that models knowledge about supply chains in software systems, to solve the problem of insecure dependencies. Second, ontologies can be reasoned upon to check for consistency and to infer implicit knowledge. If the qualitative factors that are computed by decision trees in prioritization models are formalized as relations with vulnerabilities in their range and datatypes in their domain, then we can let a reasoner figure out if a vulnerability should be patched or not. [3] Third, ontologies support semantic-based searches, as opposed to textual-based searches. This makes it easier to report and retrieve information, since there are no terminological confusions. In the rest of the paper we will show how our system, TRONTO, achieves these goals.

Natural language processing techniques offer another approach to overcome the current limitations of the NVD. In particular, many concepts and relations that are not yet represented in the NVD can be added to an ontological representation with the help of NLP tools. For instance, Jacobs et al. build a classifier that uses the referenced documents in selected CVE entries as a training corpus to be able to predict which vulnerabilities will have actual exploits [17]. This is a very suggestive approach. We will show that, by associating CVEs with texts from different sources (CVE references, but also tweets, social media posts, etc.), we can build categorized corpora that have strong predictive power, and can assign CVEs to their CWE class, score them on the CVSS index, and more. That is, we can leverage the knowledge that analysts have already stored in the NVD to automate the analysis of future CVEs, reducing the amount of effort on the part of analysts and solving the bottleneck problem. Ultimately, we would like for vulnerability management teams to be able to extract the information they need directly from the stream of online text. The ontology would run in the background, serving the purpose of finding semantically relevant documents. NLP tools come in at this point to query the documents, also helping security management teams in researching vulnerabilities. Importantly, for practitioners this would mean that additional and more timely information is made available to them. Linguistic models and dashboards can be employed to integrate the information and make the input to practitioners more natural.

3 ONTOLOGIES, THE SEMANTIC WEB, AND CYBERSECURITY

From a narrow point of view, the Semantic Web is an extension of the World Wide Web that enhances the human-readable text of web pages with metadata to make their content understandable to machines. This is made possible with the help of formal languages that represent the meaning of the data so that it can be reasoned upon, and must therefore be based on a semantically-interpretable logical model. Such languages are used to build knowledge representations of the content found in the web, i.e. ontologies. But this view of the Semantic Web as an artifact is giving way to a broader understanding of the term, as a field of study that is "about establishing efficient (that is, low cost) methods and tools for data sharing, discovery, integration, and reuse" [14]. The methods developed for the Semantic Web (the artifact), then, can be applied beyond the World Wide Web to achieve efficient management and integration of information, with substantial gains over more traditional data-management methods.

One advantage that Semantic Web technologies have over other forms of knowledge representation (such as databases) is their ability to integrate information from multiple and often heterogeneous sources, without human intervention [14, 16]. The WWW allows for the sharing and integration of information by means of hyperlinked pages, but this information is text-based and unstructured, requiring humans to process it. To describe web resources in a unified, non-local manner, and the relations among them, the semantic web makes use of the Resource Description Framework (RDF) language. When the resources identified in RDF annotations are given meaning in the form of defined terms, an RDF graph is turned into an ontology, "a model of (some aspect of) the world [which] introduces vocabulary describing various aspects of the domain being modeled and provides an explicit specification of the meaning of that vocabulary" [16]. But understanding meaning is much more than mapping text onto a hierarchy of well-defined

[3] An alternative to decision trees is to use first order Boolean logic. The OWL formalism (a standard for the creation of ontologies) has the power to construct restricted classes, which have as their instances those individuals satisfying a logical constraint. A disadvantage of this formalism is that it may be difficult for laypeople to understand. But if the categorization is left to the reasoner then humans only need to deal with the output. We leave it to future research to figure out how to model decision trees in this way.

terms and relations, it also involves making inferences based on common assumptions and other forms of implicit knowledge. The Ontology Web Language (OWL) extends the capabilities of RDF with a version of first-order logic (Description Logic) that has constructors for complex classes (e.g. Boolean connectors, quantifiers, transitive relations, cardinality constraints) and supports reasoning. Ontologies (and their cousins linked data and knowledge graphs) are the artifacts that make it possible for machines to understand the content of web pages, but they do more than that. Ontologies can be also applied to the output of social media exchanges (e.g. Twitter), and it lends itself to very efficient semantic-based queries (using SPARQL) with significant advantages over text-based methods.

The security model behind the NVD is based on two major conceptual classes: configurations (i.e. applications and operating systems) and vulnerabilities. The NVD aggregates, analyzes, and classifies vulnerabilities that are discovered and disclosed by various parties. Each verified individual vulnerability is given a CVE number as an identifier. A vulnerability is, first and foremost, a bug in a piece of code which puts the system running the code (the configuration) at risk of being targeted by a cyberattack. Configurations, then, have vulnerabilities. This conceptual model lends itself nicely to be formalized into an ontology, with two core classes and a core property relating them. We have built these major classes (with their subclasses and instances) in TRONTO. The ontology, which is written directly as an OWL document, models the relations between two major taxonomic domains: one for `configurations` of software and hardware and another one for `vulnerabilities`. To build the ontology, we have "lifted" the structured information that already exists in two expert sources: the NVD and NIST's Common Platform Enumeration (CPE). NVD uses a slice of the Common Weaknesses Enumeration (CWE) to classify vulnerabilities[4]. This slice forms the taxonomic backbone of the Vulnerabilities portion of the ontology (with an additional class for unclassified vulnerabilities: NVDCWEnoinfo). Individual vulnerabilities (reported in the Common Vulnerabilities and Exploits, CVE) are the instances. CPE, on the other hand, offers a dictionary with Individual Resource Identifiers (IRI) for a vast array of computer systems and components[5]. The major segments of an IRI are `part`, `vendor`, and `product`. `Part` classifies configurations into `application`, `operating system`, and `hardware`, and gives us the first taxonomic layer of the configurations half of the ontology. Products are the instances of this part of the ontology. Products may have other attributes (e.g. vendor, language, etc.) which we may choose to model as separate classes (even though the vendor is a higher-order attribute than product, a product is a configuration, but a vendor is not).

The two databases (NVD and CPE) are cross-linked so that for every vulnerability there is a list of affected configurations. This allows us to define a non-taxonomic property "has_vulnerability", which takes a Configuration as its domain and a Vulnerability as its range (we also define an inverse relation "is_vulnerability_of"). There are two datatype properties, "has_severity_level" and "has_description", which link a vulnerability to its severity level and its description, respectively. The description is a string, and the severity level is a point in a scale from 0 to 4 (4 being "critical")[6]. Figure 1 illustrates the components of the ontology.

4 EXPANDING THE REACH OF THE NVD THROUGH ONTOLOGICAL REPRESENTATIONS

TRONTO's content and structure are coextensive with that of the NVD. So it is fair to ask what is gained by representing the knowledge stored in the NVD as an ontology. In this section we will discuss three ways in which a semantic representation, like the one we get from an ontology, exceeds the functionality of a database. This will show how the Semantic Web can improve on the current state of our cybersecurity tools.

The first extension comes from the use of a reasoner to infer more knowledge than what is explicitly stated in the database. OWL gets more of its expressive power from a form of first-order logic called Description Logic (DL). A reasoner like Pellet or HermiT will evaluate the DL axioms of the ontology and it will infer new statements, which will be added to the knowledge representation (a reasoner may also find that the ontology is inconsistent). For example, vulnerabilities are assigned a severity level in the NVD, but applications are not directly evaluated for their safety. If an IT technician wants to know whether an application has critical vulnerabilities, for instance, all she can do is search for the vulnerabilities the application has, and check each one for its severity score. This is a tedious and time-consuming task. But because the information is already there, in a way, we can use the expressive power of OWL to infer it from the classes and relations in the ontology. OWL allows for the creation of restricted classes, that is, classes that are defined by logical restrictions placed on them. For instance, we can define a class of all entities that have the property "hasSeverityLevel" with a value of 3 (i.e. high). When conjoined with the class of vulnerabilities, this results in a new class for vulnerabilities with a high severity level. We do the same for critical vulnerabilities. The code below illustrates how the classes are defined in OWL:

- class: SeverityHighVulnerability:
  equivalentTo Vulnerability and hasSeverityLevel value 3
- class: SeverityCriticalVulnerability:
  equivalentTo Vulnerability and hasSeverityLevel value 4

The membership in these classes does not need to be specified by hand. A reasoner will take the information already existent in the ontology (the fact that each vulnerability is assigned a severity level) and use it to make an inference that allows it to assign the vulnerability to the right class. But the work of the reasoner does not stop there. We have also created a restricted class for the configurations, such that it will contain all configurations that have a vulnerability classified as CRITICAL or HIGH in severity. Now, if a member of a vulnerability management team, a developer checking for security issues in upstream libraries, or any other user wants to know if an application is worthy of trust (because its code does

[4]https://nvd.nist.gov/vuln/categories
[5]https://nvd.nist.gov/products/cpe

[6] CVEs are associated with a baseScore and a baseSeverity features. CVSS scores are mapped to qualitative severity ratings in the following way: 0.0 = None, 0.1-3.9 = Low, 4.0-6.9 = Medium, 7.0-8.9 = High, 9.0 - 10.0 = Critical. To handle our data internally, we use a simple scale from 0 (= None) to 4 (= Critical).

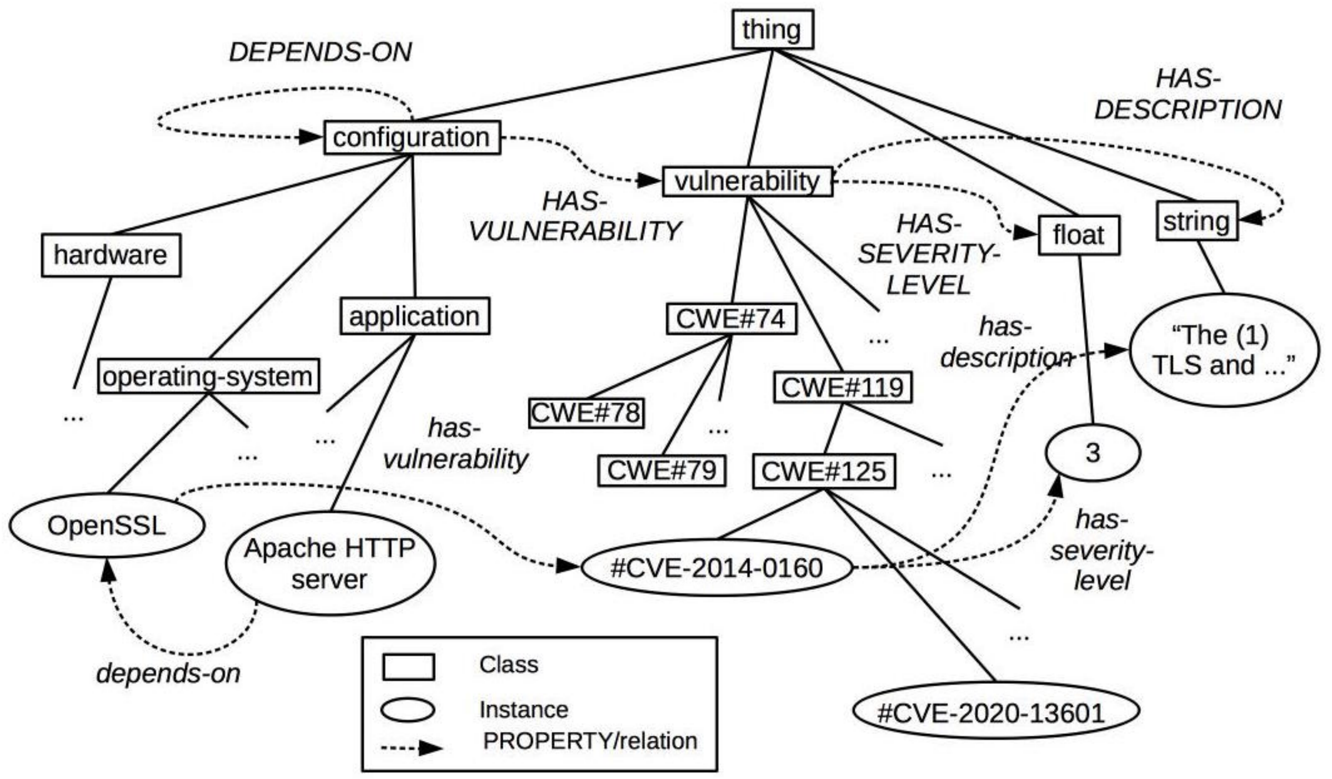

Figure 1: Diagram of the TRONTO ontology. The ontology provides a formal representation of expert knowledge. It defines relationship between vulnerabilities and dependencies collected from structured information.

not contain severe vulnerabilities), all she needs to do is query the ontology to see which class the application has been placed under by the reasoner.

- class SevereRiskConfiguration:
  equivalentTo Configuration and
  hasVulnerability some
    SeverityCriticalVulnerability or
    SeverityHighVulnerability or

There is a second way in which we can use the power of the ontological representation to enrich the content of the NVD. CVE entries link the vulnerabilities in the NVD database to the systems affected, which are given a CPE identifier. An IT technician or software maintainer, then, can find out which vulnerabilities their system has by performing a simple word-based search. But more often than not the safety of an application is a function of the vulnerabilities that affect its dependencies, information that is not part of the NVD or the CPE. The vulnerabilities that an application inherits from its dependencies, then, will remain hidden from view from the user consulting the NVD, contributing to a false sense of security [5, 33]. In a recent study, Prana et al. [24] sampled 450 projects using open-source libraries to analyze their vulnerabilities, looking for their types, distribution, severity, and persistence in a period of one year. They obtained dependency information about the projects with an industrial software composition analysis tool. Among their results is the finding that most dependency vulnerabilities persist throughout their observation period, and that it takes 3-5 months to fix the resolved ones. What this shows is that the problem of insecure libraries remains a thorny issue that current tools are unable to solve.

Representing knowledge about vulnerabilities in an ontology like TRONTO may hold the key to solving the problem of insecure libraries. First, gathering information about dependencies is notably hard. They can be gathered from manifest files in some GitHub repositories, or by commercial tools like the Veracode Software Composition Analysis tool used in [24], but the reality is that there is no central repository where dependencies can be found. In any case, it is possible to enrich the "configuration" branch of the ontology with dependency annotations, specifying a "depends_on" relation between software products. This can be done manually, by human curators of the ontology, or interactively by the end-users: one way our system collects dependency information is by asking an end-user that is interested in checking the safety of an application to enter a list of the application's dependencies. Over time, the edges connecting products will grow. A query to the ontology, then, checks not only if the application of interest has vulnerabilities, but also whether its dependencies have them.

Moreover, the issue of the insecure libraries is a cumulative one, since dependencies have their own dependencies. Some systems have the ability to report back on the chain of dependencies, but not all do. In OWL, it is easy to define a relation as "transitive", and use this feature of the relation when information is retrieved. Once the ontology has all the instances and their relations, we can check to see if a product is vulnerable by searching through its network of dependencies. To that effect, we define a function that targets an application and collects all of its dependencies using the transitive nature of the "depends_on" property by way of the INDIRECT prefix in OWL. This function generates a list of all the dependencies of a configuration, closed under transitivity, and checks to see if any of the dependencies, or the app itself, are in the domain of a "has_vulnerability" relation with a vulnerability in its range. The ontology and the associated functions are implemented in a web-based query system that can be used to find out the severity level of an application.

A third way in which an ontology-based approach to security goes beyond the coverage of the NVD is by expanding the concepts and relations in it. Besides vulnerabilities and configurations, security is also concerned with threats, countermeasures, and other concepts and relations. The meanings of these terms, and the relations that link one to the other, are defined in "risk models" of security. Some projects have elaborated ontologies for cybersecurity based on these models. Välja et al. [32] propose to automate threat modeling for critical infrastructure, but they use knowledge graphs instead of OWL. Razzaq et al. [26] propose an Ontology for attacks, while Guo and Wang [12] develop an Ontology to model CVEs (similar to our own TRONTO). Shepard et al. [29] develop an ontology that organizes information collected from a computer network, in combination with general information about threats and vulnerabilities. The purpose of the ontology (and the system it supports) is to investigate possible avenues for attack on a particular network.

In another recent proposal, Booth and Turner develop a Vulnerability Description Ontology (VDO) which, like TRONTO, builds on the structured information collected in the NVD. [3] The VDO incorporates interesting empirical insights from actual security response teams, but in our opinion it is too embryonic to constitute a true ontology. The VDO takes CVEs as the central unit, without integrating them into a true taxonomy (CWE categories, which we take to be taxonomic concepts in TRONTO, are represented as optional attributes of the CVEs). While the VDO does not commit to a formalism, ours adheres to current standards by adopting the OWL representation, and its foundation in Description Logic. This makes it possible for our ontology to be reasoned upon and checked for consistency, which their ontology is not able to do. Moreover, in addition to its ontology, TRONTO gathers information from both technical and non-technical sources using linguistic methods to analyze language, and allows queries. In other words, it provides a system rather than just an ontology.

For our purposes, we find the work of Herzog et al. [13] very useful. They develop an ontology for cybersecurity, named "Security", based on concepts from a risk-analysis model. The core classes are "Vulnerability", "Asset", "Threat", and "Countermeasure", with "Security_Goal" and "Defence_Strategy" as additional concepts. A diagram of their ontology is shown in Figure 2:

Herzog et al. [13]'s Security ontology is written in OWL, and is available on the web at https://www.ida.liu.se/divisions/adit/security/projects/secont/Security.owl. One of the advantages of OWL ontologies is that they can be modified and reused, so OWL includes functions to import the concepts and relations from one ontology into another one. We have expanded the contents of TRONTO by importing the classes and relations from the Security ontology. With this setup, TRONTO + Security ontology can be used to aggregate structured knowledge from another database: the Common Attack Pattern Enumeration and Classification (CAPEC). CAPEC provides "a comprehensive dictionary of known patterns of attack employed by adversaries to exploit known weaknesses in cyber-enabled capabilities". CAPEC does for the "Threat" class of the Security ontology what CWE does for the "Vulnerability" class of the TRONTO ontology. Moreover, the Security ontology includes a class of "Asset", which is roughly equivalent to our "Configuration" class. Assets may include the diverse set of software and hardware (e.g. servers, workstations, network devices) that an organization needs to run in order to conduct business. These assets may be decentralized, as it happens in a large health services organization with laboratories running diagnostics, clinical trials, etc. Or diversified financial services institutions with many branches. As part of their Asset Management program, an IT office collects information about the state of the assets, including any security risks, and deploys patches and other mitigation (counter)measures according to security policies [31]. Asset Management Systems use the most recent information in the CVE database to scan a system for vulnerabilities. We would like to design our ontology to have points of contact with Asset Management Systems to provide a similar functionality.

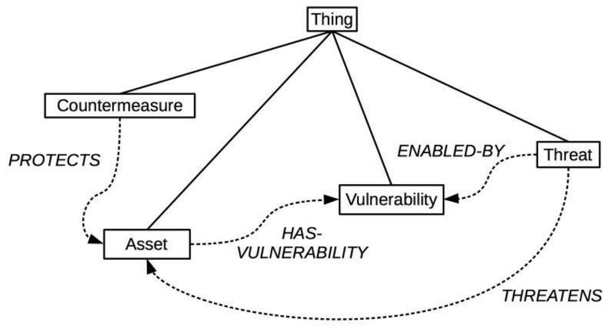

Figure 2: Core concepts and relations in the Security ontology

Now we present a case study about the advantage of using our ontology. Heartbleed (CVE-2014-0160) was a serious vulnerability in the popular OpenSSL cryptographic library, a widely used implementation of the Transport Layer Security (TLS) protocol. This vulnerability allows attackers to steal the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. This vulnerability has a severity score of 7.5, and is wide-spread across the world. According to an estimate [8], 24–55% of HTTPS servers in the Alexa Top 1 Million websites were initially vulnerable to this attack. Apache server, one of the most popular web server software that are used to host websites, are vulnerable to this vulnerability as it employs OpenSSL.

So imagine a company is hosting its websites on the Apache server, and the IT technician wants to measure how secure it is. Searching “Apache” on the CVE website is a good idea, but not enough, as it doesn’t check whether the dependencies of the Apache server have vulnerabilities. Querying the CVE website about all direct/transitive dependencies of Apache server is a doable task, but it is extremely time-consuming for the IT technician since Apache server has a large number of direct/transitive dependencies. On the other hand, if she uses our ontology by just searching “Apache”, our ontology will automatically fetch all the direct/transitive dependencies of Apache web server since the ontology already has information about the dependency on OpenSSL. Then, the ontology will check whether any of those dependencies are vulnerable and, if the severity level of any of those is high or critical (as it is the case with CVE-2014-0160), it will infer that Apache server is a high-risk application. Finally, it will return the result to the user in an instant.

Another benefit of using a Semantic Web approach for cases like the Heartbleed virus is that ontologies force the research community to clarify terminological issues. For instance, does "Heartbleed" refer to a vulnerability, or to a threat? If threats and vulnerabilities are disjoint classes, and the unique instance "Heartbleed" is a member of both classes, the ontology would be rendered inconsistent. On the other hand, there are many terms for the same object: "CVE-2014-0160", "Heartbleed", "Heartbeat", and "SOL15159". In an ontological representation this issue is easily resolved, since all the names map onto the same global resource.[7]

5 FINDING CYBERSECURITY INFORMATION OUTSIDE THE NVD

In the previous section we discussed three ways in which an ontology, which is the artifact at the core of the Semantic Web, can extend the capabilities of a database like the NVD when we try to represent expert knowledge about cybersecurity, namely: (1) providing additional information that is not explicit in the database but can be inferred from existing information; (2) enriching the contents of the database through structure that connects pieces of information in a meaningful way, for example, supporting transitivity in relationships; and (3) providing a framework to relate additional information (e.g. threats, countermeasures) in a structured way. But the real power of the Semantic Web is in the integration of heterogeneous sources of information. In this section we discuss our efforts to leverage our ontology to search for and query social media threads that contain cybersecurity-related information. Because the ontology contains not just concepts and relationships, but also links to documents with descriptions and other relevant information about the concepts in the ontology, it serves not only as a source of information for human consumption, but as data from which data-driven models can be built. Using natural language processing techniques, these models can relate unstructured information in the form of text from websites and social media to concepts in the ontology. These models can help identify relevant information as it appears, and this new information can then be added to the ontology, possibly after human review.

NVD does an excellent job of reporting and disseminating information about vulnerabilities, once they are discovered and analyzed, but there may be a lag between the time of discovery and the time of reporting [28, 38]. Moreover, while browsing through the CVE entries may be useful for cybersecurity experts, it does not seem to be the best way for developers and users to gain knowledge about vulnerabilities [23]. Users and developers like to get advice on the safety of their systems and their security practices from the web and from social media. Stack Overflow, Security Stack Exchange, and Reddit are some of the hubs in which people can post their cybersecurity-related inquiries on[2, 35].

There is a growing body of literature on how to detect social media exchanges related to cybersecurity, Tweets in particular. Many recent approaches built classifiers for automatic mining of cybersecurity-related information from social platforms. The performance of current methods greatly improves on early attempts at classifying cybersecurity-related social media posts, like using keyword matching. For example, Lippman et al. [19] built Logistic Regression (LR) and Support Vector Machine (SVM) based binary classifiers to extract cybersecurity-related posts from Stack Exchange, Reddit, and Twitter, and Dionísio et al. [7] utilized a Convolutional Neural Network (CNN) to detect cybersecurity-related tweets on Twitter. These approaches leverage advances in text processing to provide information and potentially reduce the workload of human analysts through automation.

Recent work has raised the possibility that data-driven models built from existing information about past vulnerabilities may be able to forecast the severity of a just-discovered vulnerability from the chatter about it. This could alleviate information overload and reduce the workload for analysts by providing a semi-automated way to identify emerging threats. For instance, Zong et al. [38] trained CNN and LR classifiers to predict the severity of a threat based on the perceived sentiment of the tweet describing that threat. The annotating process for the training data works as follows: For each tweet, Mechanical Turkers annotated whether it may contain cybersecurity threats. If a tweet is labeled with threats, the threats are then marked as “severe” or “not severe” according to the perceived sentiment of the tweet. Since the annotators are not security experts, their judgements are more based on their limited domain knowledge towards the sentiment, or opinion, of the tweets. Even though the authors found that using tweets can accurately forecast cybersecurity threats at least five days before the threats are published in the NVD database, their research did not conclusively show that perceived sentiments collected by annotators align with expert analysis reflecting the real severity level of the issue. Figure 4 shows how well their model based on sentiment (“opinion”) predicts high severity of a threat as rated by experts. While the model is effective in its most confident predictions, as shown by the high precision at 10, 20 and 50, it is substantially less precise as more tweets are considered. As a proof-of-concept of analysis of new unstructured information based on data already available,

[7]The reverse of the synonymy problem is the ambiguity (or polysemy) problem, when the same word may map onto two distinct (and maybe related) terms or concepts. The double sense of “Heartbleed” as either a vulnerability or a threat is an example of a general kind of polysemy inherent in discussions about security. Linguists have developed lexical ontologies, such as WordNet [21] [9] to deal with these issues. A reviewer points out that the term “vulnerability” may be vague, since two analysts may have different concepts about what a vulnerability is and how to define it. The danger of unresolved ambiguities is that it may lead to inconsistencies in the ontology, so this is a problem that deserves further scrutiny.

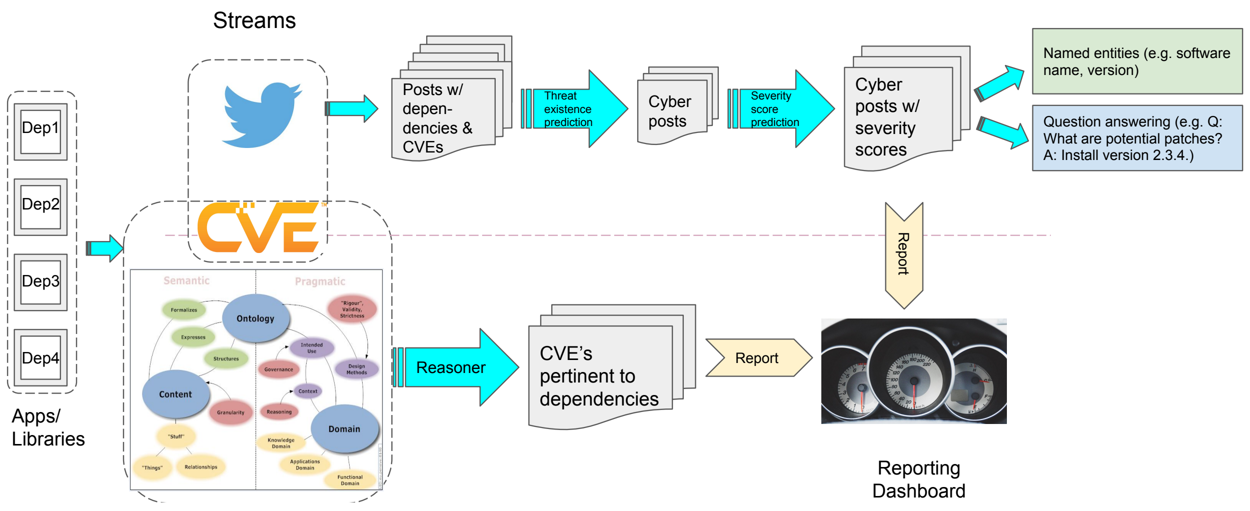
Streams
Dep1
Dep2
Dep3
Dep4
Apps/ Libraries
Posts w/ depen- dencies & CVEs
Threat existence prediction
Cyber posts
Severity score prediction
Cyber posts w/ severity scores
Named entities (e.g. software name, version)
Question answering (e.g. Q: What are potential patches? A: Install version 2.3.4.)
Report
Semantic
Pragmatic
Ontology
Formalizes
Expresses
Structures
Content
Granularity
"Stuff"
"Things"
Relationships
"Rigour", Validity, Strictness
Intended Use
Governance
Design Methods
Context
Reasoning
Domain
Knowledge Domain
Applications Domain
Functional Domain
Reasoner
CVE's pertinent to dependencies
Report
Reporting Dashboard

Figure 3: TRONTO pipeline for analyzing vulnerabilities for software and dependencies. On the bottom, we utilize structured information embedded in our ontology. On the top, we extract information such as entities and predict severity level from related tweets. Both sources of information are integrated and reported to users in a dashboard.

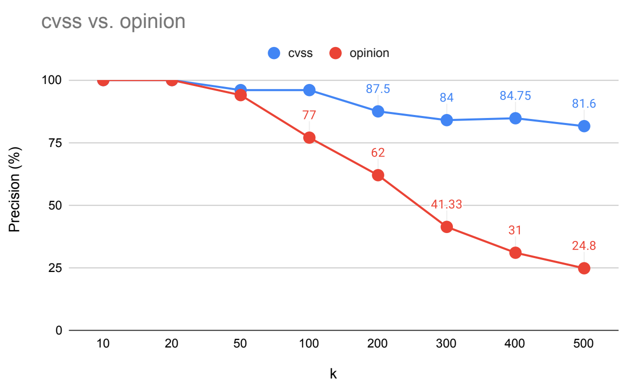

Figure 4: Performance comparison against opinion-based prediction from previous work of identifying severe threats with precision at k.

we instead related tweets mentioning vulnerabilities to their (now known) severity scores. The main difference is the use of CVSS as the training signal, instead of non-expert sentiment associated with tweets. Our results (Figure 4, "cvss") suggest that assignment of severity scores for CVEs can be approximated by fully automatic means. Given the imperfect nature of predictions based on the text of tweets alone, there is still uncertainty associated with these predictions when going beyond the top 200 tweets the model is most confident about, but this uncertainty is substantially diminished compared to the model trained on non-expert opinion.

We have also examined the possibility of automatic categorization of CVEs based on textual descriptions. In our experiments, we used CVEs from two common types, CWE-79 (Improper Neutralization of Input During Web Page Generation) and CWE-119 (Improper Restriction of Operations within the Bounds of a Memory Buffer), and using a dataset of over 3,000 CVEs, we found that these CVEs can be automatically categorized based only on their textual descriptions nearly perfectly (with approximately 99% accuracy for CVEs not used to build the classification model).

In its current state, our system integrates information from social media, which provides more up-to-date, and sometimes more detailed information, to enhance the knowledge base in our ontology. In response to a query about the vulnerabilities that affect a configuration, our system deploys a list of relevant CVEs, but it does more than that: it finds a relevant set of cybersecurity-related tweets, and displays them to the user with more in-depth analysis.

Specifically, we process through the following pipeline to utilize information from tweets.

- Retrieve tweets related to the software and its dependencies that we are interested in from the last seven days.
- Run a threat existence classifier to find if a tweet is related to cybersecurity vulnerability.
- Run a severity forecast model to predict severity scores on tweets predicted as threats.
- Extract entities from the tweets using a named entity recognition tagger.
- Integrate the tweets sorted with severity prediction scores into the ontology and display tweets, distribution of CVE numbers, and severity levels.
- Deploy a question-answering system so that users can easily find information from linked web pages.

When a TRONTO query is made, the system uses the Twitter API to retrieve tweets that are related to the affected software and its dependencies. These tweets, however, are not necessarily about cybersecurity issues. To filter out the tweets that are not relevant to cybersecurity, we train a classifier (initialized with BERT [6]) on a corpus of 4,800 examples from Zong et al. [38] which were hand-tagged as to whether they discussed a threat or not. The classifier achieves an accuracy of 82% on the development set (1200 examples).

Once the cybersecurity-related tweets are segregated from the rest, we forecast the severity of the threat. Following previous research, we assume that a CVSS score larger than 7.0 is severe, otherwise not severe. We train a classifier (initialized with BERT) on a set of tweets which are linked to CVEs with actual severity scores. We get reliable results with a training sample of 1,500 tweets. Training a text classifier initialized with BERT, we compared the performance of prediction by training on actual CVSS scores and annotated opinions. Evaluated by precision@k [38], we found that with the same number of training samples (1,500 tweets), opinion-based severity prediction is not reliable (Figure 4). In comparison, Our system runs the cybersecurity-related tweets through the classifier, and then sorts them by their predicted scores (in the reverse order) to be displayed to the user.

Most previous work focuses on monitoring social media such as Twitter to detect and classify potential threats and alert users. However, we found that most filtered tweets, which provide concise messages regarding possible cybersecurity threats, also contain URLs that may link to additional detailed information such as blogs and news reports. The additional information includes affected use cases, steps to reproduce the issue, and potential patches suggested by the community. Such details are not specified in social media or CVE databases, but can be very beneficial to users who can quickly find promising solutions. To this end, we integrate an extractive question-answering system based on a distilled version of BERT [27, 34] trained with SQuAD [25]. For a tweet with links to external information, users can ask questions such as "what software is affected" and "is there a patch" to get answers easily, if these are found in the text extracted from these links.

6 LIMITATIONS

In its current form, then, the NVD faces a couple of serious challenges: Its metrics may be out of step with the needs of the users, and the analysts are overwhelmed by the sheer number of vulnerabilities they need to review. Our system proposes to mitigate these shortcomings by a) formalizing the knowledge built into the NVD as an ontology that can be reasoned upon and queried directly and b) augmenting the knowledge in the NVD with a stream of information emanating from heterogeneous sources (i.e., social media). The danger in this approach, however, is to undermine some of the advantages afforded by a curated database like the NVD: the authoritative nature of the content, and the trustworthiness of the sources of information.

A potential limitation of our approach, then, is that our system does not perform any vetting of the external sources it reports or uses for trustworthiness. But for that reason, and for safety's sake, TRONTO currently reports the tweets it finds back to the initiator of the query. Going forward, we foresee some built-in mechanism to ascertain trustworthiness, perhaps using some of the media-internal available metrics (e.g., reputation score in StackOverflow, or some kind of social network analysis of centrality of the writer of certain tweets). [8] Social media platforms are coming up with ways for the community to police the information that is shared through them. But we have seen that social media can be used to spread misinformation, often with malicious intent. A system that relies on community-based sources of information is susceptible to manipulation in the same way that social media is. A mature version of our system would need to include safeguards against malicious misinformation, and even against adversarial intrusion into the system itself. During development, we debated whether to use the queries themselves to serve as input to the system (adding information about applications, vulnerabilities, and dependencies, for instance). We decided against it, precisely because of the risk of inserting noisy or unreliable information that could turn the ontology inconsistent, and break down the reasoner. Until we figure out a workable solution, opening up the system to external sources in this way may enable an adversarial user to suppress information or even exploit the system to "push" a specific product or patch, with negative consequences.

The goal of a knowledge representation system like TRONTO is not to replace the authoritative sources (i.e., the NVD), but to augment them by linking to other sources of information. One potential limitation of our approach is that a user will assume that the system is in a closed, final state. But that is not the case. By having the tweets and other sources of information presented in a transparent way, the user can still engage in the usual social media exchanges (re-tweeting information, contributing to online discussions, or contacting the authors of tweets or posts directly for further clarification, for instance). Our system leverages the power of NLP Q&A methods to cut through time-consuming searches, automating routine tasks to reduce the load on the users.

The target users of the system can be grouped into two classes: a) security analysts and vulnerability hunters, whose goal is to discover, categorize, and disseminate information about vulnerabilities, and b) developers and system administrators who are interested in keeping programs free of vulnerabilities by adopting best practices when writing code or applying patches. A limitation of our system is that it can't just replace an analyst or a developer, in part because of the issues mentioned above. It is true that we are proposing to emulate the work of an analysts based on a language model. But our model will be as trustworthy as the source documents used for training. Moreover, our system can't yet replace the human element in the analysis of reported vulnerabilities. But that is not our aim: TRONTO can be used as a support system to simplify the job of the analyst. For developers, the danger is that having a system that runs in parallel to their work may actually increase development time without a commensurable increase in safety. Ideally, the ontology would run in the background, alerting the developer when potential vulnerability has been introduced.

7 DISCUSSION

Over the last two decades, a dedicated cadre of cybersecurity experts set out to collect and organize the resources they needed to root out malicious code and protect against nefarious attacks. The result was a small group of databases like the NVD, containing

[8] https://stackoverflow.com/help/whats-reputation, https://meta.stackoverflow.com/questions/251487/getting-to-know-stack-overflows-voting-culture

reliable information about weaknesses, which were overseen by experts in the field. At the same time, discussions about cybersecurity continued to proliferate on social media. And while the databases were a good source of information for investigations about cybersecurity and for testing hypotheses about threats and vulnerabilities, many users continued to rely on the advice they received in sites like StackOverflow when they needed to evaluate the safety of their systems.

There is a trade-off between the agility afforded by social media discussions and the reliability of expertly-curated databases. Social media may even be a few steps ahead than the databases in predicting the emergence of the next vulnerability. The challenge ahead is to integrate these heterogeneous sources of information in a single knowledge-representation model. In this paper we have shown that the technologies developed for the Semantic Web, with an OWL ontology at its center, can do this for cybersecurity, in the same way that it has been done for the biomedical sciences.[9]

For this enterprise to succeed, it is reasonable to start with the "low hanging fruit", lifting the structured information in curated databases like the NVD, CVE, CPE, or CAPEC to provide the classes, relations, and instances of the ontology. Social media streams, on the other hand, contain unstructured information, and a different toolkit needs to be employed to integrate those sources. Fortunately, the Semantic Web has also been developing techniques to retrieve information from textual corpora, leveraging the semantic resources provided by OWL ontologies. In the road ahead, we will explore the benefits of substituting semantic-based methods to find cybersecurity-related tweets and to query social media sources for the text-based methods we tested here.

[9]Standards like SNOMED CT and the like, which resolve issues around medical terminology, contain useful lessons for the development of ontologies for cybersecurity. In fact, many cybersecurity concepts, like that of a "computer virus", are metaphorical extensions of biomedical concepts. One important difference between the two fields, however, is that in cybersecurity we assume the existence of an adversarial party that is actively trying to exploit weaknesses in a system.

REFERENCES

[1] Afsah Anwar, Ahmed Abusnaina, Songqing Chen, Frank Li, and David A. Mohaisen. 2020. Cleaning the NVD: Comprehensive Quality Assessment, Improvements, and Analyses. *CoRR* abs/2006.15074 (2020). arXiv:2006.15074 https://arxiv.org/abs/2006.15074

[2] Shahab Bayati and Marzieh Heidary. 2016. Information security in software engineering, analysis of developers communications about security in social q&a website. In *Pacific-Asia Workshop on Intelligence and Security Informatics*. Springer, 193–202.

[3] Harold Booth and Christopher Turner. 2016. *Vulnerability description ontology (vdo): a framework for characterizing vulnerabilities.* Technical Report. National Institute of Standards and Technology.

[4] Robert Byers, David Waltermire, Christopher Turner, et al. 2020. *Collaborative Vulnerability Metadata Acceptance Process (CVMAP) for CVE Numbering Authorities (CNAs) and Authorized Data Publishers.* Technical Report. National Institute of Standards and Technology.

[5] Alexandre Decan, Tom Mens, and Eleni Constantinou. 2018. On the Impact of Security Vulnerabilities in the Npm Package Dependency Network. In *Proceedings of the 15th International Conference on Mining Software Repositories* (Gothenburg, Sweden) *(MSR '18)*. Association for Computing Machinery, New York, NY, USA, 181–191. https://doi.org/10.1145/3196398.3196401

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[7] Nuno Dionísio, Fernando Alves, Pedro M Ferreira, and Alysson Bessani. 2019. Cyberthreat detection from twitter using deep neural networks. In *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[8] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, et al. 2014. The matter of heartbleed. In *Proceedings of the 2014 conference on internet measurement conference*. 475–488.

[9] Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database.* MIT Press, Cambridge, MA.

[10] First.org. 2019. Computer Security Incident Response Team (CSIRT) Services Framework Version 2.1. https://www.first.org/standards/frameworks/csirts/FIRST_CSIRT_Services_Framework_v2.1.0.pdf/ [Online; posted November-2019].

[11] Assane Gueye and Peter Mell. 2021. A Historical and Statistical Studyof the Software Vulnerability Landscape. *arXiv preprint arXiv:2102.01722* (2021).

[12] Minzhe Guo and Ju An Wang. 2009. An ontology-based approach to model common vulnerabilities and exposures in information security. In *ASEE Southest Section Conference*.

[13] Almut Herzog, Nahid Shahmehri, and Claudiu Duma. 2007. An ontology of information security. *International Journal of Information Security and Privacy (IJISP)* 1, 4 (2007), 1–23.

[14] Pascal Hitzler. 2021. A review of the semantic web field. *Commun. ACM* 64, 2 (2021), 76–83.

[15] Hannes Holm and Khalid Khan Afridi. 2015. An expert-based investigation of the common vulnerability scoring system. *Computers & Security* 53 (2015), 18–30.

[16] Ian Horrocks. 2008. Ontologies and the semantic web. *Commun. ACM* 51, 12 (2008), 58–67.

[17] Jay Jacobs, Sasha Romanosky, Idris Adjerid, and Wade Baker. 2020. Improving vulnerability remediation through better exploit prediction. *Journal of Cybersecurity* 6, 1 (2020), tyaa015.

[18] Yuning Jiang, Manfred A. Jeusfeld, and Jianguo Ding. 2021. Evaluating the Data Inconsistency of Open-Source Vulnerability Repositories. *The 16th International Conference on Availability, Reliability and Security* (2021).

[19] Richard P Lippman, David J Weller-Fahy, Alyssa C Mensch, William M Campbell, Joseph P Campbell, William W Streilein, and Kevin M Carter. 2017. Toward finding malicious cyber discussions in social media. In *AAAI Workshops*.

[20] Peter Mell. 1999. *Understanding the world of your enemy with I-CAT (Internet-Categorization of Attacks Toolkit).* Technical Report. NATIONAL INST OF STANDARDS AND TECHNOLOGY GAITHERSBURG MD COMPUTER SECURITY DIV.

[21] George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM Vol. 38, No. 11* (1995), 39–41.

[22] Éamonn Ó Muirí. 2019. Framing software component transparency: Establishing a common software bill of material (SBOM). (2019).

[23] Tayyaba Nafees, Natalie Coull, Ian Ferguson, and Adam Sampson. 2017. Idea-Caution Before Exploitation: The Use of Cybersecurity Domain Knowledge to Educate Software Engineers Against Software Vulnerabilities. 133–142. https://doi.org/10.1007/978-3-319-62105-0_9

[24] Gede Artha Azriadi Prana, Abhishek Sharma, Lwin Khin Shar, Darius Foo, Andrew E Santosa, Asankhaya Sharma, and David Lo. 2021. Out of sight, out of mind? How vulnerable dependencies affect open-source projects. *Empirical Software Engineering* 26, 4 (2021), 1–34.

[25] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, 2383–2392. https://doi.org/10.18653/v1/D16-1264

[26] Abdul Razzaq, Zahid Anwar, H Farooq Ahmad, Khalid Latif, and Faisal Munir. 2014. Ontology for attack detection: An intelligent approach to web application security. *computers & security* 45 (2014), 124–146.

[27] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR* abs/1910.01108 (2019). arXiv:1910.01108 http://arxiv.org/abs/1910.01108

[28] Clemens Sauerwein, Christian Sillaber, Michael M Huber, Andrea Mussmann, and Ruth Breu. 2018. The tweet advantage: An empirical analysis of 0-day vulnerability information shared on twitter. In *IFIP International Conference on ICT Systems Security and Privacy Protection*. Springer, 201–215.

[29] Blake Shepard, Cynthia Matuszek, C. Bruce Fraser, William Wechtenhiser, David Crabbe, Zelal Güngördü, John Jantos, Todd Hughes, Larry Lefkowitz, Michael Witbrock, Doug Lenat, and Erik Larson. 2005. A knowledge-based approach to network security: Applying Cyc in the domain of network risk assessment. In *IN PROC. OF THE 17TH INNOVATIVE APPLICATIONS OF ARTIFICIAL INTELLIGENCE CONFERENCE*. AAAI Press AAAI Press / The MIT Press, 1563–1568.

[30] Jonathan M Spring, Allen Householder, Eric Hatleback, Art Manion, Madison Oliver, Vijay Sarvapalli, Laurie Tyzenhaus, and Charles Yarbrough. 2021. *Prioritizing Vulnerability Response: A Stakeholder-Specific Vulnerability Categorization (Version 2.0).* Technical Report. CARNEGIE-MELLON UNIV PITTSBURGH PA.

[31] Michael Stone, Chinedum Irrechukwu, Harry Perper, Devin Wynne, and Leah Kauffman. 2018. *IT Asset Management.* Technical Report. NATIONAL INST. OF STANDARDS AND TECHNOLOGY.

[32] Margus Välja, Fredrik Heiding, Ulrik Franke, and Robert Lagerström. 2020. Automating threat modeling using an ontology framework. *Cybersecurity* 3, 1 (2020), 1–20.

[33] Jeff Williams and Arshan Dabirsiaghi. 2012. The Unfortunate Reality of Insecure Libraries. (2012). https://www.scribd.com/document/175866686/Aspect-Security-the-Unfortunate-Reality-of-Insecure-Libraries

[34] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6

[35] Xin-Li Yang, David Lo, Xin Xia, Zhi-Yuan Wan, and Jian-Ling Sun. 2016. What security questions do developers ask? a large-scale study of stack overflow posts. *Journal of Computer Science and Technology* 31, 5 (2016), 910–924.

[36] Emrah Yasasin, Julian Prester, Gerit Wagner, and Guido Schryen. 2020. Forecasting IT security vulnerabilities–An empirical analysis. *Computers & Security* 88 (2020), 101610.

[37] Su Zhang, Xinming Ou, and Doina Caragea. 2015. Predicting cyber risks through national vulnerability database. *Information Security Journal: A Global Perspective* 24, 4-6 (2015), 194–206.

[38] Shi Zong, Alan Ritter, Graham Mueller, and Evan Wright. 2019. Analyzing the Perceived Severity of Cybersecurity Threats Reported on Social Media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 1380–1390. https://doi.org/10.18653/v1/N19-1140