information

Article

Tracing CVE Vulnerability Information to CAPEC Attack Patterns Using Natural Language Processing Techniques

Kenta Kanakogi [1,*], Hironori Washizaki [1], Yoshiaki Fukazawa [1], Shinpei Ogata [2], Takao Okubo [3], Takehisa Kato [4], Hideyuki Kanuka [4], Atsuo Hazeyama [5] and Nobukazu Yoshioka [6]

1 Department of Computer Science and Engineering, Waseda University, Shinjuku-ku, Tokyo 169-8555, Japan; washizaki@waseda.jp (H.W.); fukazawa@waseda.jp (Y.F.)
2 Institute of Engineering, Academic Assembly, Shinshu University, Nagano City, Nagano 380-8553, Japan; ogata@cs.shinshu-u.ac.jp
3 Institute of Information Security, Yokohama, Kanagawa 221-0835, Japan; okubo@iisec.ac.jp
4 Hitachi, Ltd., Chiyoda-ku, Tokyo 100-8280, Japan; takehisa.kato.wx@hitachi.com (T.K.); hideyuki.kanuka.dv@hitachi.com (H.K.)
5 Department of Information Science, Tokyo Gakugei University, Koganei-shi, Tokyo 184-8501, Japan; hazeyama@u-gakugei.ac.jp
6 Research Institute for Science and Engineering, Waseda University, Shinjuku-ku, Tokyo 169-8555, Japan; nobukazuy@acm.org
* Correspondence: kanakogi-soft@fuji.waseda.jp

Abstract: For effective vulnerability management, vulnerability and attack information must be collected quickly and efficiently. A security knowledge repository can collect such information. The Common Vulnerabilities and Exposures (CVE) provides known vulnerabilities of products, while the Common Attack Pattern Enumeration and Classification (CAPEC) stores attack patterns, which are descriptions of common attributes and approaches employed by adversaries to exploit known weaknesses. Due to the fact that the information in these two repositories are not linked, identifying related CAPEC attack information from CVE vulnerability information is challenging. Currently, the related CAPEC-ID can be traced from the CVE-ID using Common Weakness Enumeration (CWE) in some but not all cases. Here, we propose a method to automatically trace the related CAPEC-IDs from CVE-ID using three similarity measures: TF–IDF, Universal Sentence Encoder (USE), and Sentence-BERT (SBERT). We prepared and used 58 CVE-IDs as test input data. Then, we tested whether we could trace CAPEC-IDs related to each of the 58 CVE-IDs. Additionally, we experimentally confirm that TF–IDF is the best similarity measure, as it traced 48 of the 58 CVE-IDs to the related CAPEC-ID.

Keywords: security knowledge repository; CVE; CAPEC; natural language processing; sentence embeddings; TF–IDF; Universal Sentence Encoder; Sentence-BERT

check for updates

Citation: Kanakogi, K.; Washizaki, H.; Fukazawa, Y.; Ogata, S.; Okubo, T.; Kato, T.; Kanuka, H.; Hazeyama, A.; Yoshioka, N. Tracing CVE Vulnerability Information to CAPEC Attack Patterns Using Natural Language Processing Techniques. *Information* **2021**, *12*, 298. https://doi.org/10.3390/info12080298

Academic Editor: Aneta Poniszewska-Maranda

Received: 1 July 2021
Accepted: 25 July 2021
Published: 26 July 2021

1. Introduction

Due to the sheer volume, system administrators spend a lot of time dealing with vulnerabilities. In order to effectively respond and mitigate vulnerabilities, vulnerability information must be collected efficiently and quickly. Additionally, the vulnerability and the attack techniques must be understood. For example, when assessing the severity and priority of vulnerabilities, it is essential to refer to information about known vulnerabilities and attacks.

To collect such information, knowledge repositories on cybersecurity issues may be used. Public repositories include Common Vulnerabilities and Exposures (CVE) [1] and Common Attack Pattern Enumeration and Classification (CAPEC) [2]. CVE lists common identifiers for known vulnerability information. CAPEC is a dictionary of common identifiers for attack patterns employed by adversaries to exploit weaknesses.

A vulnerability scanner such as Vuls (https://vuls.biz/, accessed on 16 June 2021) can automatically detect CVE-IDs, but CVEs do not contain rich information about an attack. Therefore, CAPEC attack patterns are used to add attack information. Due to the fact that CVE and CAPEC are not directly linked, it is difficult to identify related CAPEC attack information from CVE vulnerability information, especially for people without security knowledge.

Currently, Common Weakness Enumeration (CWE) [3], which is a list of common identifiers of types of security weaknesses, is employed to identify the relationships between CVE and CAPEC. "Weakness Enumeration" contains information about the relationship between the CVE vulnerability information and CWE. "Related Attack Patterns" includes the CAPEC attack pattern information related to the CWE information. Via CWE, it is possible to trace the related CAPEC-ID from the CVE-ID. In this paper, we refer to this method as the "conventional method". There are two issues with the conventional method, which previous studies [4–8] have yet to solve:

- When using CWE, some patterns cannot trace from the CVE-ID to the related CAPEC-ID. Sections 3 and 5.2 provide specific cases and conditions;
- The linking between CVE–CWE and CWE–CAPEC is carried out manually. Manual linking cannot handle the growing amount of vulnerability information. In addition, more link failures may occur.

Currently, CVE and CAPEC are not explicitly linked. The creator of CVE-ID should link it directly to CAPEC. However, accurate linking is costly and difficult. In this paper, we aim to trace related CAPEC-IDs directly from CVE-IDs. Our method recommends which CAPEC-IDs should be linked to a given CVE-ID (this paper is an extension of a paper presented at the Hawaii International Conference on System Sciences (HICSS 54) [9]. Here, we expanded the natural language processing technique and revised the analysis results as well as the corresponding discussions), but it does not provide a definitive optimal CAPEC-ID. We measure the similarity between the CAPEC documents and the CVE description.

We propose three approaches to calculate similarity: TF–IDF [10], Universal Sentence Encoder (USE) [11], and Sentence-BERT (SBERT) [12]. We chose three representative similarity measures. These measures can be categorized into two groups: word count-based and inference-based methods. The former includes TF–IDF and are simple methods that calculate similarity based on the frequency of occurrence of words in a document. The latter includes USE and SBERT. They create models that can gain distributed representations of words and sentences and then calculate the similarity. Previous studies have employed similarity measures [4], [5], but they have not been accurately evaluated or directly compared. By tracing 58 CVE-IDs to related CAPEC-IDs, we compare these three approaches to the conventional method. Our approach is described in detail in Section 4.

This paper aims to answer the following three research questions (RQs):

RQ1. When tracing relationships between security repositories, how accurately can it be traced from CVE-ID to CAPEC-ID? This question researches the tracing accuracy of CVE-ID to CAPEC-ID;

RQ2 When using a similarity measurement based on natural language processing techniques, how accurately can CVE-IDs be traced to CAPEC-IDs? This question confirms the usefulness of our proposed approach;

RQ3 Which of the three evaluated methods provides the best results? This question identifies the most useful method among the three methods proposed in RQ2.

The contributions of this paper are threefold:

1. We elucidate the linking accuracy between CVE–CWE and CWE–CAPEC;
2. Our method can easily identify CAPEC-IDs that are link candidates and assist in the linking process;
3. The person reporting the vulnerability information can determine whether the report contains sufficient security information.

This paper is organized as follows. Section 2 introduces related works. Section 3 presents a motivating example. Section 4 explains our approach. Section 5 presents the results of the experiment and discusses the RQs. Section 6 provides the conclusions and future work.

2. Related Work

Previous studies have investigated the relationship between repositories [4–8]. One study [4] employs TF–IDF to associate CVE with CAPEC, but only assesses Software Defined Networking and Network Functions Virtualization vulnerability. Additionally, [5] had a similar objective to our study. They employed Doc2vec, but did not explicitly assess accuracy.

Another study automatically mapped CVE to CAPEC and ATT&CK in order to find appropriate mitigation [6]. They created a neural network model with automatic classification in an attempt to realize a deep learning model that groups CWE into CAPEC. However, they only suggested a method. On the other hand, this study conducts experiments with 58 CVE-IDs and prepares the correct CAPEC-ID.

Some study papers [13–15] use topic modeling and natural language processing in order to extract hidden topics from the textual description of each attack pattern and learn topic models. Although we performed a simple natural language process, the topic analysis performed in these other papers is a useful reference for future applications of topic analysis.

The literature on the application of similarity measures is available in [16,17]. In particular, the paper [16] proposes a hybrid method that combines TF–IDF-weighted Doc2Vec and TF–IDF-weighted VSM.

Other researches have studied vulnerability ontology models [18–20]. They have investigated vulnerability models based on famous security knowledge repositories, such as CVE, CWE, and CAPEC. These studies use security knowledge repositories to analyze and assess risk and security [21–26]. However, they do not describe the process to obtain information from this security knowledge repository. The studies [27–31] focus on mining methods and information retrieval for security knowledge repositories. In these papers, relationships were used to mine information from each repository. However, they do not regard the accuracy of the repository relationships as a problem. We explicitly evaluate the accuracy of the relationships between repositories.

In the Vulnerability Assessment and Penetration Test process [32–34], our method can suggest attack patterns to penetration testers. The authors of [34] used CVE and CAPEC to perform penetration testing. Their method uses the relationships among knowledge repositories. The study does not regard the accuracy of the relationships in the repository as a problem.

3. Motivating Example

Via CWE, it is possible to trace some of the related CAPEC-IDs from the CVE-ID, but not all patterns can be traced. An example is CVE-2018-18442. This CVE-ID is vulnerability information about network packet flooding. The description of CVE-2018-18442 is as follows:

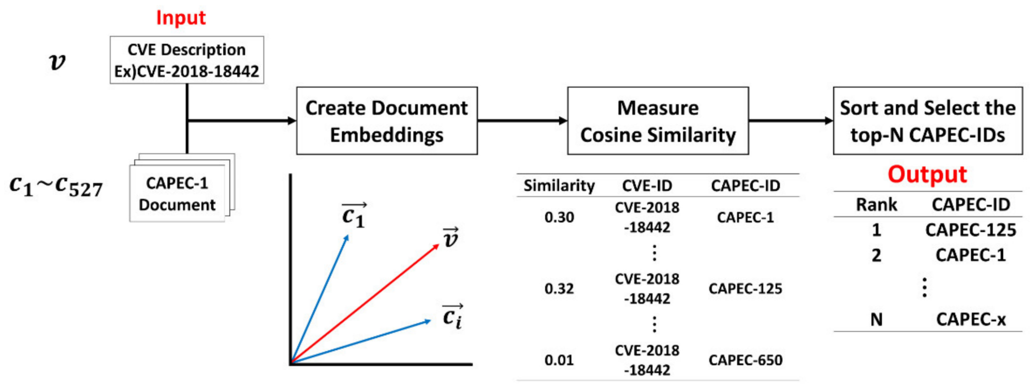
> *D-Link DCS-825L devices with firmware 1.08 do not employ a suitable mechanism to prevent denial-of-service (DoS) attacks. An attacker can harm the device availability (i.e., live-online video/audio streaming) by using the hping3 tool to perform an IPv4 flood attack. Verified attacks includes SYN flooding, UDP flooding, ICMP flooding, and SYN-ACK flooding.* (https://cve.mitre.org/cgi-bin/cvename.cgi?name=2018-18442, accessed on 16 June 2021)

CAPEC-125 is an identifier that summarizes attack information about flooding. CVE-2018-18442 is linked to CWE-20. CWE-20 and CAPEC-125 are not linked. Therefore, it is not possible to trace CAPEC-125 from CVE-2018-18442. Tracing the CVE–CWE–CAPEC relationship, we found that the correct CAPEC-ID could not be identified. The issue arises with the use of CWE. Therefore, it is preferable to trace directly from CVE to CAPEC. This is our motivation.

4. Tracing Method from CVE-ID to CAPEC-ID

We propose a method that enables direct tracing from CVE to CAPEC (Figure 1). Our method has four steps. First, we collect a CVE-ID description and 527 CAPEC-ID documents in a corpus. For example, CVE-2018-18442 is used as the input data in Figure 1. Second, we use natural language processing techniques to create document embeddings. Third, the cosine similarity shown in Equation (1) is calculated for the CVE-ID vector and the vector of all CAPEC-IDs. Finally, the CAPEC documents are sorted by the calculated similarity scores; N CAPEC-IDs are selected. As a result, the related CAPEC-IDs are traced from a CVE-ID.

$$\cos\left(\vec{p}\cdot\vec{q}\right) = \frac{p_1q_1 + p_2q_2 + \cdots + p_nq_n}{\sqrt{{p_1}^2 + {p_2}^2 + \cdots + {p_n}^2}\sqrt{{q_1}^2 + {q_2}^2 + \cdots + {q_n}^2}} = \frac{\vec{p}\cdot\vec{q}}{\left|\vec{p}\right|\left|\vec{q}\right|} \quad (1)$$


Input
v
CVE Description Ex)CVE-2018-18442
$c_1 \sim c_{527}$
CAPEC-1 Document
Create Document Embeddings
Measure Cosine Similarity
Sort and Select the top-N CAPEC-IDs
$\vec{c_1}$
$\vec{v}$
$\vec{c_i}$
Output

Similarity	CVE-ID	CAPEC-ID
0.30	CVE-2018 -18442	CAPEC-1
⋮	⋮	⋮
0.32	CVE-2018 -18442	CAPEC-125
⋮	⋮	⋮
0.01	CVE-2018 -18442	CAPEC-650

Rank	CAPEC-ID
1	CAPEC-125
2	CAPEC-1
⋮	⋮
N	CAPEC-x

Figure 1. Overview of our method.

We investigated three methods to create document embeddings: TF–IDF, USE, and SBERT. We describe each approach in detail below. All the approaches are represented by a flow using CVE-2018-18442 as the input data to find the related CAPEC-ID.

4.1. Tracing Method Based on TF–IDF

TF–IDF evaluates the importance of words in a document and is mainly used in the fields of information retrieval and topic analysis. TF–IDF is calculated based on two indices: TF (term frequency) and IDF (inverse document frequency). The TF–IDF score is the product of TF and IDF.

TF is the frequency that a word appears in a document. It is given in Equation (2). $tf(t,d)$ is the TF value of some word t in document d. $n_{t,d}$ is the number of occurrences of a word t in document d. $\sum_{s\in d} n_{s,d}$ means the sum of the number of occurrences of all words in document d.

$$tf(t,d) = \frac{n_{t,d}}{\sum_{s\in d} n_{s,d}} \quad (2)$$

IDF is the reciprocal of the document frequency in which a word occurs. Words that appear in many documents are unlikely to be characteristic words in one document, which is expressed as Equation (3). The logarithm is used to ensure that the value remains stable as the number of documents increases. $idf(t)$ is the IDF value of a word t, N is the number of all documents, and $df(t)$ is the number of documents in which a word t appears.

$$\text{idf(t)} = \log\frac{N}{df(t)} \quad (3)$$

The words are extracted from each document. Then a vector of TF is created and IDF is multiplied by all the words. This vector is used to calculate the cosine similarity between two documents.

We use scikit-learn [35]. Herein the "TfidfVectorizer" method is employed. "TfidfVectorizer" converts each document into a vector based on the TF–IDF score. Figure 2 shows the flow of the TF–IDF approach using CVE-2018-18442 as an example.

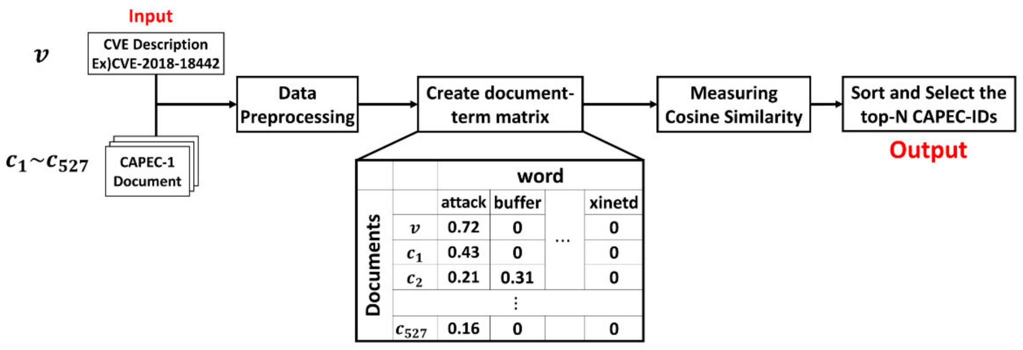


Figure 2. Overview of the tracing method based on TF–IDF.

The algorithm of the TF–IDF approach is as follows:

STEP 1: Input data. Input all CAPEC descriptions and the description of one CVE-ID as a corpus.

STEP 2: Preprocess data. Preprocess the corpus with the most common words, removing punctuation, tokenization, and lemmatization.

STEP 3: Create a document–term matrix using "TfidfVectorizer" and "fit_transform". "TfidfVectorizer" converts a collection of corpora into a matrix with 5602 TF–IDF features, while "fit_transform" learns the matrix of the TF–IDF features and returns the TF–IDF-weighted document–term matrix.

STEP 4: Calculate the cosine similarity. We calculate the cosine similarity between a CVE-ID and all CAPEC-IDs using the TF–IDF-weighted document–term matrix.

STEP 5: Sort the similarity. STEP 4 produces the similarity between the inputted CVE-ID (CVE-2018-18442) and all CAPEC-IDs. The scores are sorted in descending order, and the ID with the higher rank is the related CAPEC-ID.

4.2. Tracing Method Based on USE

USE is an encoder model that converts a sentence with an arbitrary length into a 512-dimensional vector. It was developed by researchers at Google and published on the "Tensorflow Hub" in 2018. The pre-trained model is available here [36]. Thus, we can easily gain a document-distributed representation by downloading the model. They proposed a model architecture for two encoder models. One is based on a deep averaging network (DAN), and the other is based on a transformer architecture. Both models take a word, sentence, or paragraph as an input and provide a 512-dimensional vector as an output. The transformer model is more accurate than the DAN model. However, the transformer model is more complex and uses more resources compared to the DAN model. We use DAN because the transformer is also used in SBERT. Since CVE and CAPEC are written in English, we used a model that can handle English. Figure 3 shows the flow of the USE approach using CVE-2018-18442 as an example. We use a pretraining model called "universal-sentence-encoder-v4" (https://tfhub.dev/google/universal-sentence-encoder/4, accessed on 16 June 2021).

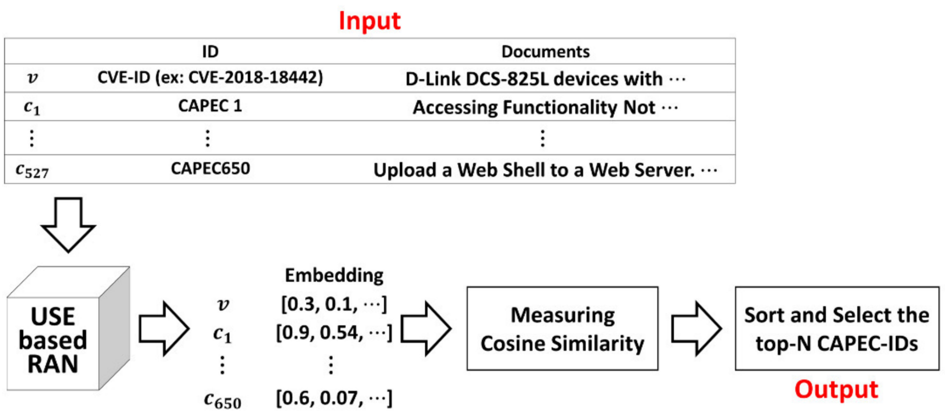

Input
ID
Documents
v CVE-ID (ex: CVE-2018-18442) D-Link DCS-825L devices with ⋯
c_1 CAPEC 1 Accessing Functionality Not ⋯
c_{527} CAPEC650 Upload a Web Shell to a Web Server. ⋯
USE based RAN
Embedding
v [0.3, 0.1, ⋯]
c_1 [0.9, 0.54, ⋯]
c_{650} [0.6, 0.07, ⋯]
Measuring Cosine Similarity
Sort and Select the top-N CAPEC-IDs
Output

Figure 3. Overview of the tracing method based on Universal Sentence Encoder.

4.3. Tracing Method Based on Sentence-BERT

SBERT generates higher-quality sentence vectors by fine-tuning BERT. SBERT is built on top of BERT. Adding a pooling layer to the pre-trained BERT model produces a high accuracy of sentence embedding. When inputting a sentence into BERT, a context-sensitive vector is outputted for each token of the input sentence. The role of pooling is to combine these vectors into a fixed-length sentence vector.

SBERT fine-tunes using a Siamese network. The Siamese network is a method that uses two neural networks to compute an embedding representation, and then compares the two embedding representations. The loss function plays an important role when fine-tuning. There are 13 types of loss functions (https://www.sbert.net/docs/training/overview.html#loss-functions, accessed on 16 June 2021). The appropriate loss function depends on the training data and the task. Here, we employ the "CosineSimilarityLoss", which consists of two texts and a float label.

We use the link information from CVE and CWE for the training data. The CWE has a field called Observed Examples. In this field, there are CVE-IDs and a text that summarizes the CVE description. Therefore, two texts, the CVE description and the CWE summary text, are prepared and trained as a pair.

Figure 4 shows the flow of the SBERT approach using CVE-2018-18442 as an example. We used a pretraining model called "paraphrase-mpnet-base-v2" (https://public.ukp.informatik.tu-darmstadt.de/reimers/sentence-transformers/v0.2/, accessed on 16 June 2021).

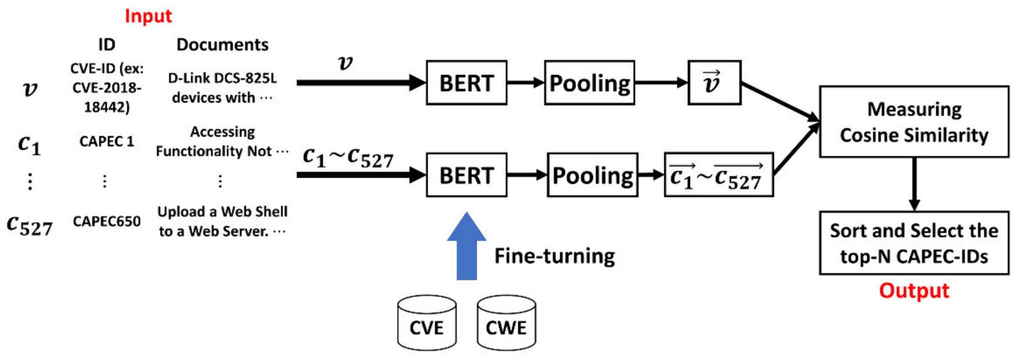

Input
ID
Documents
v CVE-ID (ex: CVE-2018-18442) D-Link DCS-825L devices with ⋯
c_1 CAPEC 1 Accessing Functionality Not ⋯
c_{527} CAPEC650 Upload a Web Shell to a Web Server. ⋯
v
BERT
Pooling
$\vec{v}$
$c_1 \sim c_{527}$
$\vec{c_1} \sim \vec{c_{527}}$
Fine-turning
CVE
CWE
Measuring Cosine Similarity
Sort and Select the top-N CAPEC-IDs
Output

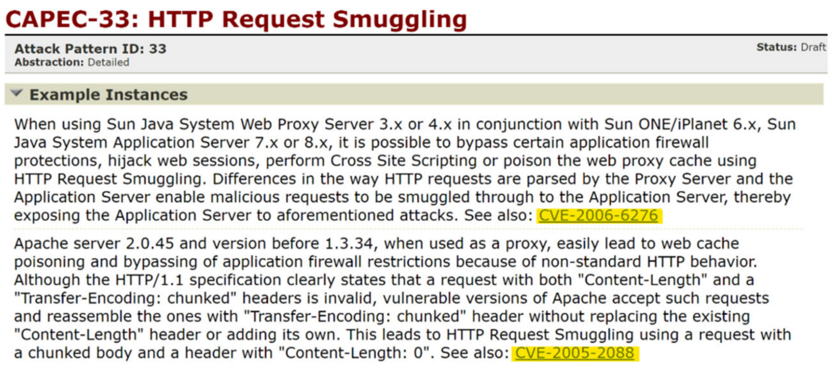
Figure 4. Overview of the tracing method based on Sentence-BERT.

5. Experiments and Results

We prepared 58 CVE-IDs and used them as input data. Then, we verified that the related CAPEC-IDs could be traced from each of the 58 CVE-IDs.

5.1. Fifty-Eight CVE-IDs

CAPEC contains the Example Instance field. In this field, the CVE-ID may be used to describe an example attack or exploit. Figure 5 shows an example, where CVE-2006-6276 and CVE-2005-2088 are listed in the Example Instance of CAPEC-33.



CAPEC-33: HTTP Request Smuggling

Attack Pattern ID: 33
Abstraction: Detailed
Status: Draft

Example Instances

When using Sun Java System Web Proxy Server 3.x or 4.x in conjunction with Sun ONE/iPlanet 6.x, Sun Java System Application Server 7.x or 8.x, it is possible to bypass certain application firewall protections, hijack web sessions, perform Cross Site Scripting or poison the web proxy cache using HTTP Request Smuggling. Differences in the way HTTP requests are parsed by the Proxy Server and the Application Server enable malicious requests to be smuggled through to the Application Server, thereby exposing the Application Server to aforementioned attacks. See also: CVE-2006-6276

Apache server 2.0.45 and version before 1.3.34, when used as a proxy, easily lead to web cache poisoning and bypassing of application firewall restrictions because of non-standard HTTP behavior. Although the HTTP/1.1 specification clearly states that a request with both "Content-Length" and a "Transfer-Encoding: chunked" headers is invalid, vulnerable versions of Apache accept such requests and reassemble the ones with "Transfer-Encoding: chunked" header without replacing the existing "Content-Length" header or adding its own. This leads to HTTP Request Smuggling using a request with a chunked body and a header with "Content-Length: 0". See also: CVE-2005-2088

Figure 5. CAPEC-33 web page (https://capec.mitre.org/data/definitions/33.html, accessed on 16 June 2021).

There are a total of 58 CVE-IDs in the Example Instance field (Appendix A). The three CVE-IDs are duplicated, but the input data are recognized as different. Therefore, we assume that the link from CVE to CAPEC is many-to-one. The average word count for the 58 CVE-IDs is 38. The median is 34. We used these 58 CVE-IDs because we need the correct data to evaluate our method's experimental results correctly. If a CVE-ID listed in the Example Instance is input, we verify that it can trace to the corresponding CAPEC-ID.

5.2. RQ 1. When Tracing Relationships between Security Repositories, How Accurately Can It Be Traced from CVE-ID to CAPEC-ID?

When tracing the relationship between CVE–CWE and CWE–CAPEC, we successfully traced 3 out of 58 CVE-IDs. The relationship between CVE and CWE is the reason for the low accuracy. We analyzed the accuracy of the link between CVE and CWE.

CVE has a URL of National Vulnerability Database. The NVD has a section called "Weakness Enumeration". This section may contain the corresponding CWE-ID for a CVE-ID. However, it may be written as "NVD-CWE-Other" or "NVD-CWE-noinfo". In these cases, CVE is not linked to CWE and cannot be traced to CAPEC. Figure 6 shows the percentage of CVE-IDs that are not linked to the CWE. A total of 28% of CVE-IDs are not linked to CWE-IDs.

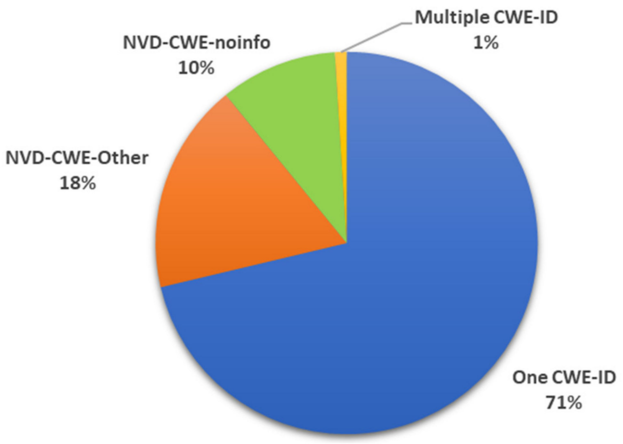


Figure 6. Percentage of CVE-IDs not linked to CWE.

When aggregated by year, the percentage of CVE-IDs linked to CWE-IDs has increased dramatically since 2008 (Figure 7). Since the percentage of CVE-IDs linked to CWE-IDs is increasing, it is important to link CVE-IDs to CWE-IDs accurately. However, bias is an issue. Only 210 of the 918 CWE-IDs are linked to CVE-IDs. We studied the distribution of the top 15 CWE-IDs among the linked CWE-IDs (Figure 8).

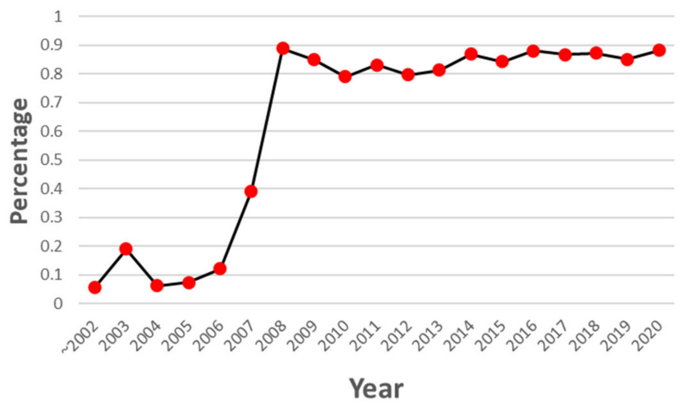


Figure 7. Changes in the percentage of CVE-ID that is linked to CWE-ID.

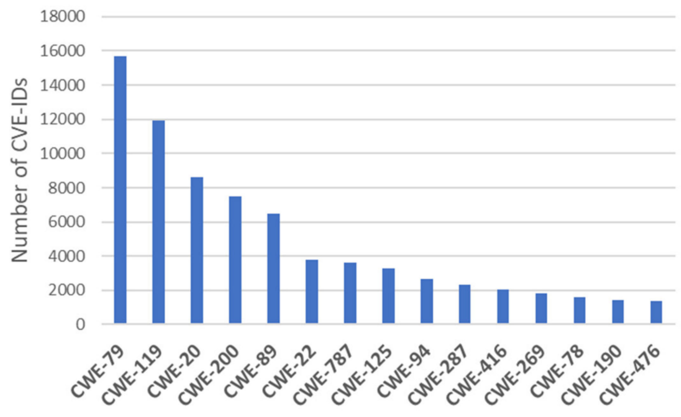


Figure 8. Vulnerability distribution by CWE-ID.

We focused on CWE-20 among the 15 CWE-IDs. CWE-20 is an identifier that indicates weaknesses related to "Improper Input Validation". CWE-20 is the parent node of the path traversal, buffer error, XSS, and injection. It covers a lot of weaknesses. The reason is described on the CWE-20 webpage:

> *The "input validation" term is extremely common, but it is used in many different ways. In some cases its usage can obscure the real underlying weakness or otherwise hide chaining and composite relationships.*
>
> *Some people use "input validation" as a general term that covers many different neutralization techniques for ensuring that input is appropriate, such as filtering, canonicalization, and escaping. Others use the term in a more narrow context to simply mean "checking if an input conforms to expectations without changing it."* (https://cwe.mitre.org/data/definitions/20.html, accessed on 16 June 2021)

From the above, we found that the CWE-20 covered a lot of weaknesses. The ability to cover many weaknesses can be linked to many vulnerabilities. Therefore, it is often linked to the CVE-ID. In addition to CWE-20, there are other such CWE-IDs. CVE-ID, which is linked to CWE for this reason, cannot provide useful vulnerability information.

As described in Section 3, we introduced a CVE-ID that cannot be traced to CAPEC-ID. CVE-2014-0160 is another example. The description of CVE-2014-0160 is as follows:

The (1) TLS and (2) DTLS implementations in OpenSSL 1.0.1 before 1.0.1g do not properly handle Heartbeat Extension packets, which allows remote attackers to obtain sensitive information from process memory via crafted packets that trigger a buffer over-read, as demonstrated by reading private keys, related to d1_both.c and t1_lib.c, aka the Heartbleed bug. (https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2014-0160, accessed on 16 June 2021)

CAPEC-540 is an identifier that summarizes buffer overread attack information. CWE-126 is an identifier for buffer overread weaknesses. However, CVE-2014-0160 is linked to CWE-119 (buffer errors). Hence, CWE-119 cannot be traced to CAPEC-540. Why is it linked to CWE-119? CVE-2014-0160 is related to multiple weaknesses in CWE-125, 126, and 130. These three CWE-IDs were recorded as identifiers related to the Heartbleed bug. Despite this fact, we believe that the CWE-119 linking is because it allows for a single identifier to indicate the presence of multiple weaknesses. Hence, we found that the CWE-ID described in Weakness Enumeration does not characterize an attacker's use of a vulnerability to attack.

Box 1. RQ 1. Answer.

RQ 1. When tracing relationships between security repositories, how accurately can it be traced from CVE-ID to CAPEC-ID? **Only 3 out of the 58 CVE-IDs were traced to the related CAPEC-ID. The accuracy of the link between CVE and CWE is not good. Therefore, the CAPEC attack information that can be traced from CVE may not be useful information**.

5.3. When Using a Similarity Measurement Based on Natural Language Processing Techniques, How Accurately Can CVE-IDs Be Traced to CAPEC-IDs?

Figures 9–11 show the experimental results for each of the three similarity measures. In this experiment, we assume that the link between CVE and CAPEC is many-to-one. The condition that there is only one ground truth is the cause of the low precision. For future work, we can improve precision by handling the somewhat less related CAPEC-ID as a ground truth.

In predicting potential attacks from vulnerabilities, we should not miss highly related attacks. Therefore, it is important to trace to the most related CAPEC-IDs, even if several unrelated CAPEC-IDs are traced. Thus, we focus on Recall@n. Recall@n indicates the proportion of relevant items found in the top n recommendations. We calculated Recall@1 to Recall@10. We traced the related CAPEC-IDs from the 58 CVE-IDs using the method described in Section 4. All three methods realized a trace. With TF–IDF, more than 80% of the CVE-IDs were successful. The CVE description word count does not affect the NLP approach. One characteristic of the CVE-IDs that the NLP approach could not trace is an insufficient description of the CVE. This is attributed to a lack of security-specific words or insufficient information to hide the underlying weaknesses.

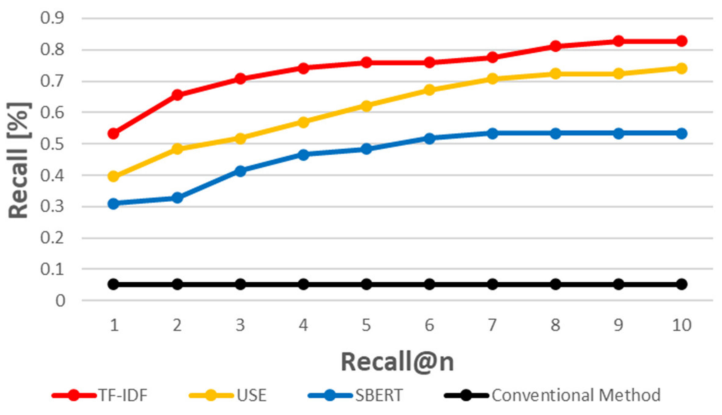


Figure 9. Recall for each approach.

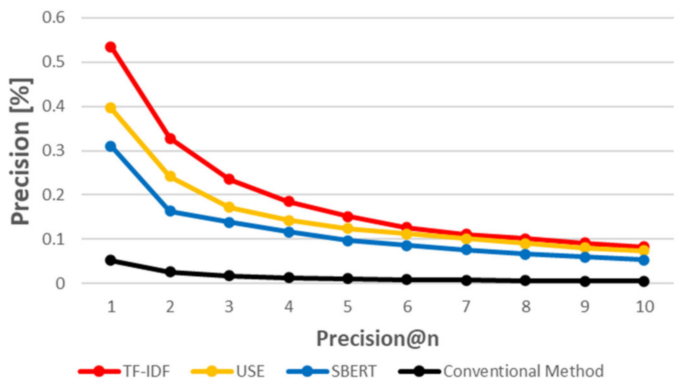


Figure 10. Precision for each approach.

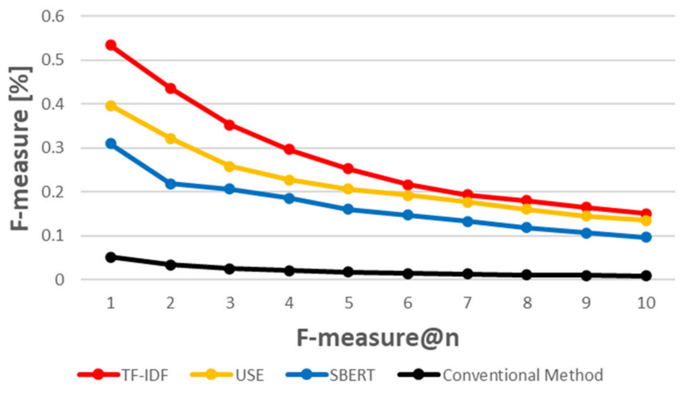


Figure 11. F-measure for each approach.

Box 2. RQ 2. Answer.

RQ 2. When using a similarity measurement based on natural language processing techniques, how accurately can CVE-IDs be traced to CAPEC-IDs? **All three methods realized a trace. TF–IDF traced 48 of the 58 CVE-IDs to the related CAPEC-ID. USE traced 43 of the 58 CVE-IDs to the related CAPEC-ID. SBERT traced 31 of the 58 CVE-IDs to the related CAPEC-ID**.

5.4. *RQ 3. Which of the Three Evaluated Methods Provides the Best Results?*

TF–IDF gave the best results (Figure 9). It traced 48 out of 58 CVE-IDs. In short, the word count-based method could trace the related CAPEC-IDs from the CVE-ID. We believe that TF–IDF had the highest recall because the corpus of TF–IDF was suitable for this task. The descriptions of CVE and CAPEC contain many system terminologies, as well as security. Therefore, many terms cannot be handled by generic pre-training models learned from wikis and other sources. On the other hand, since TF–IDF uses CAPEC as its corpus, TF–IDF can handle these terminologies well. Consequently, the accuracy of TF–IDF improved in this study. We also compared the similarity measures to each other.

5.4.1. Word Count-Based Method vs. Inference-Based Method

The word count-based method (TF–IDF) was superior. However, 10 CVE-IDs could not be traced. Two of these 10 CVE-IDs were traced by the inference-based methods, USE, and SBERT. One of them is CVE-2020-0601. CVE-2020-0601 is related to CAPEC-475. The description of CVE-2020-0601 is as follows:

> *A spoofing vulnerability exists in the way Windows CryptoAPI (Crypt32.dll) validates Elliptic Curve Cryptography (ECC) certificates. An attacker could exploit the vulnerability by using a spoofed code-signing certificate to sign a malicious executable, making it appear the file was from a trusted, legitimate source, aka 'Windows CryptoAPI Spoofing Vul-*

nerability'. (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-0601, accessed on 16 June 2021)

In the CVE description, there is "Cryptography" and "CryptoAPI". In CAPEC-475 documents, "cryptographic", "cryptographically", and "cryptanalysis" frequently occurred. TF–IDF considers these words to be mismatched. Thus, they had a low similarity. When creating the pre-training models for USE and SBERT, these words were likely learned as words with similar meanings. Thus, the tracing was successful via the inference-based method.

5.4.2. USE vs. SBERT

There are two differences between USE and SBERT. The first is the encoder. The model in USE is trained with a DAN encoder. This encoder takes the average of the variance representations of words and bigrams and feeds them into a forward propagation network to create a variance representation. Thus, word order is ignored. By contrast, the SBERT model is trained with a transformer encoder. The word order and the surrounding context are considered when generating the distributed representation. CVE is a short sentence. However, CAPEC is a document, and consequently a very long sentence. The distributed representation created by the transformer encoder contains a lot of information, which may have complicated the dependencies and syntactic structure. Therefore, DAN, which ignores word order and other factors, was more accurate.

The second is a pre-training model. For USE, we employed the model provided by "Tensorflow Hub" [36]. For SBERT, we employed the model provided by "UKPLab" [37]. It is believed that the training data were suitable for this task when the pre-trained model was created for USE. We fine-tuned the BERT pre-trained model. However, the quality of training data was poor. Hence, the accuracy could be improved using data collected by the organization or created by security experts.

Box 3. RQ 3. Answer.

RQ 3. Which of the three evaluated methods provides the best results? **TF–IDF has the highest accuracy. In other words, the word count-based method is suitable for tracing the related CAPEC-ID from the CVE-ID. To evaluate the similarity of security-specific documents, TF–IDF, which can identify security terminologies and measure their importance, gave the best results**.

5.5. Threats to Validity

Ground truth relationships between CVE-IDs and CAPEC-IDs were based on the link set by MITRE. Hence, correctness is ensured. However, there may be some links that MITRE overlooked. This is a threat to internal validity. In the future, we would like to verify not only MITRE's link but also other links.

A threat to external validity is that we experimented with only 58 CVE-IDs, but new vulnerabilities are reported daily in CVE. However, our proposed approaches are not specific to test case patterns. In the future, we plan to verify that our proposed approaches are valid for the entire CVE and to confirm their effectiveness.

6. Conclusions and Future Work

Herein we propose an approach to trace the related CAPEC-ID directly from the CVE-ID. The conventional tracing method uses the relationships between each repository. However, not only is manual tracing required, but accuracy may also be an issue. Our proposed tracing method uses similarities. The similarity between CVE-ID and CAPEC-ID is calculated using three measurements: TF–IDF, USE, and SBERT. TF–IDF has the highest accuracy, but there is still room for improvement. One possible solution for improvement is ensemble learning [38]. Ensemble learning is a technique that combines multiple learners in order to obtain better predictions. In most cases, it tends to yield better results than using a single model alone. Specifically, we can combine the predictions of the word count-based method and the inference-based method in a process such as "taking the average" or "taking the majority" of the predictions.

Our method cannot take the seriousness of each vulnerability into account yet. Additionally, we have not verified its adaptability to other security knowledge repositories, such as CWE. These themes are future work. In this paper, we referred to attack pattern information in CAPEC. However, there is other attack pattern information. An example is the pattern language [39]. We will evaluate that attack pattern information as candidates for tracing in the future. Moreover, we would like to improve the accuracy and increase the amount of information in order to provide helpful information for cybersecurity. By collecting, identifying, and analyzing data directly from security knowledge repositories, we hope to develop the proposed method into comprehensive and proactive cyber threat intelligence (CTI) research [40,41] or to extend the tracking by organizing the relationships between security concepts with a metamodel [42].

Author Contributions: Conceptualization, methodology, and writing—original draft, K.K.; funding acquisition, H.W.; formal analysis and writing—review and editing, all authors. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the SCAT Research Grant, the MEXT enPiT-Pro Smart SE: Smart Systems and Services innovative professional Education program, and the JST-Mirai Program grant number JPMJMI20B8.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data are available upon request.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. List of 58 CVE-IDs and their corresponding CAPEC-IDs.

Number	CAPEC-ID	CVE-ID	CVE Word Count
1	CAPEC-7	CVE-2006-4705	26
2	CAPEC-10	CVE-1999-0046	9
3	CAPEC-10	CVE-1999-0906	16
4	CAPEC-13	CVE-1999-0073	24
5	CAPEC-16	CVE-2003-1096	33
6	CAPEC-26	CVE-2007-1057	64
7	CAPEC-27	CVE-2000-0972	27
8	CAPEC-27	CVE-2005-0894	30
9	CAPEC-27	CVE-2006-6939	24
10	CAPEC-29	CVE-2007-1057	64
11	CAPEC-31	CVE-2010-5148	45
12	CAPEC-31	CVE-2016-0353	47
13	CAPEC-33	CVE-2005-2088	80
14	CAPEC-33	CVE-2006-6276	49
15	CAPEC-34	CVE-2006-0207	34
16	CAPEC-39	CVE-2006-0944	16
17	CAPEC-42	CVE-1999-0047	10
18	CAPEC-46	CVE-1999-0946	13
19	CAPEC-46	CVE-1999-0971	20
20	CAPEC-47	CVE-2001-0249	32
21	CAPEC-47	CVE-2006-6652	51
22	CAPEC-49	CVE-2004-1143	28
23	CAPEC-50	CVE-2006-3013	81
24	CAPEC-52	CVE-2004-0629	42
25	CAPEC-54	CVE-2006-4705	26

Table A1. *Cont.*

Number	CAPEC-ID	CVE-ID	CVE Word Count
26	CAPEC-55	CVE-2006-1058	31
27	CAPEC-59	CVE-2001-1534	41
28	CAPEC-59	CVE-2006-6969	45
29	CAPEC-60	CVE-1999-0428	14
30	CAPEC-60	CVE-2002-0258	48
31	CAPEC-61	CVE-2004-2182	26
32	CAPEC-64	CVE-2001-1335	32
33	CAPEC-66	CVE-2006-5525	55
34	CAPEC-67	CVE-2002-0412	50
35	CAPEC-70	CVE-2006-5288	29
36	CAPEC-71	CVE-2000-0884	35
37	CAPEC-72	CVE-2001-0784	26
38	CAPEC-77	CVE-2000-0860	32
39	CAPEC-80	CVE-2000-0884	35
40	CAPEC-92	CVE-2007-1544	37
41	CAPEC-93	CVE-2006-0201	34
42	CAPEC-108	CVE-2006-6799	55
43	CAPEC-135	CVE-2007-2027	41
44	CAPEC-136	CVE-2005-2301	37
45	CAPEC-267	CVE-2010-0488	42
46	CAPEC-459	CVE-2004-2761	36
47	CAPEC-459	CVE-2005-4900	54
48	CAPEC-475	CVE-2020-0601	48
49	CAPEC-632	CVE-2005-0233	50
50	CAPEC-632	CVE-2005-0234	44
51	CAPEC-632	CVE-2005-0235	44
52	CAPEC-632	CVE-2005-0236	44
53	CAPEC-632	CVE-2005-0237	47
54	CAPEC-632	CVE-2005-0238	43
55	CAPEC-632	CVE-2009-0652	81
56	CAPEC-632	CVE-2012-0584	33
57	CAPEC-657	CVE-2006-3976	16
58	CAPEC-657	CVE-2006-3977	23

References

1. Common Vulnerabilities and Exploits. Available online: https://cve.mitre.org/ (accessed on 16 June 2021).
2. Common Attack Pattern Enumeration and Classification. Available online: https://capec.mitre.org/ (accessed on 16 June 2021).
3. Common Weakness Enumeration. Available online: https://cwe.mitre.org/ (accessed on 16 June 2021).
4. Dang, Q.; François, J. Utilizing attack enumerations to study SDN/NFV vulnerabilities. In Proceedings of the 2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), Montreal, QC, Canada, 25–29 June 2018; pp. 356–361. [CrossRef]
5. Navarro, J.; Legrand, V.; Lagraa, S.; Francois, J.; Lahmadi, A.; Santis, G.D.; Festor, O.; Lammari, N.; Hamdi, F.; Deruyver, A.; et al. HuMa: A multi-layer framework for threat analysis in a heterogeneous log environment. In Proceedings of the 10th International Symposium on Foundations & Practice of Security, Nancy, France, 23–25 October 2017; pp. 144–159. [CrossRef]
6. Aghaei, E.; Shaer, E.A. ThreatZoom: Neural Network for Automated Vulnerability Mitigation. In Proceedings of the 6th Annual Symposium on Hot Topics in the Science of Security, New York, NY, USA, 1–3 April 2019; pp. 1–3. [CrossRef]
7. Scarabeo, N.; Fung, B.C.M.; Khokhar, R.H. Mining known attack patterns from security-related events. *PeerJ Comput. Sci.* **2015**, *1*, e25. [CrossRef]
8. Ma, X.; Davoodi, E.; Kosseim, L.; Scarabeo, N. Semantic Mapping of Security Events to Known Attack Patterns. In Proceedings of the 23rd International Conference on Natural Language and Information Systems, Paris, France, 13–15 June 2018; Volume 10859, pp. 91–98.
9. Kanakogi, K.; Washizaki, H.; Fukazawa, Y.; Ogata, S.; Okubo, T.; Kato, T.; Kanuka, H.; Hazeyama, A.; Yoshioka, N. Tracing CAPEC Attack Patterns from CVE Vulnerability Information using Natural Language Processing Technique. In Proceedings of the 54th Hawaii International Conference on System Sciences, Kauai, HI, USA, 4–8 January 2021; pp. 6996–7004. [CrossRef]
10. Miller, D.; Leek, T.; Schwartz, R. A hidden Markov model information retrieval system. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 15–19 August 1999; pp. 214–221. [CrossRef]

11. Cer, D.; Yang, Y.; Kong, S.Y.; Hua, N.; Limtiaco, N.; St. John, R.; Constant, N.; Guajardo-Cespedes, M.; Yuan, S.; Tar, C.; et al. Universal sentence encoder for English. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Brussels, Belgium, 31 October−4 November 2018; pp. 169–174. [CrossRef]
12. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 3–7 November 2019; pp. 3982–3992.
13. Ouchn, J.N. Method and System for Automated Computer Vulnerability Tracking. The United States Patent and Trademark Office. U.S. Patent 9,871,815, 16 January 2018.
14. Adams, S.; Carter, B.; Fleming, C.; Beling, P.A. Selecting system specific cybersecurity attack patterns using topic modeling. In Proceedings of the 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications and 12th IEEE International Conference on Big Data Science and Engineering, Trustcom, New York, NY, USA, 31 July 2018–3 August 2018; pp. 490–497. [CrossRef]
15. Mounika, V.; Yuan, X.; Bandaru, K. Analyzing CVE Database Using Unsupervised Topic Modelling. In Proceedings of the 6th Annual Conference on Computational Science and Computational Intelligence, Las Vegas, NV, USA, 5–7 December 2019; pp. 72–77. [CrossRef]
16. Ou, S.; Kim, H. Unsupervised Citation Sentence Identification Based on Similarity Measurement. In Proceedings of the 13th International Conference on Transforming Digital Worlds, Sheffield, UK, 25–28 March 2018; pp. 384–394. [CrossRef]
17. Kim, D.; Seo, D.; Cho, S.; Kang, P. Multi-co-training for document classification using various document representations: TF–IDF, LDA, and Doc2Vec. *J. Inf. Sci.* **2019**, *477*, 15–29. [CrossRef]
18. Zhu, L.; Zhang, Z.; Xia, G.; Jiang, C. Research on Vulnerability Ontology Model. In Proceedings of the 2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference, Chongqing, China, 24–26 May 2019; pp. 657–661. [CrossRef]
19. Gao, J.B.; Zhang, B.W.; Chen, X.H.; Luo, Z. Ontology-based model of network and computer attacks for security assessment. *J. Shanghai Jiaotong Univ. Sci.* **2013**, *18*, 554–562. [CrossRef]
20. Ansarinia, M.; Asghari, S.A.; Souzani, A.; Ghaznavi, A. Ontology-based modeling of DDoS attacks for attack plan detection. In Proceedings of the 6th International Symposium on Telecommunications, Tehran, Iran, 6–8 November 2012; pp. 993–998. [CrossRef]
21. Wang, J.A.; Wang, H.; Guo, M.; Zhou, L.; Camargo, J. Ranking attacks based on vulnerability analysis. In Proceedings of the 43rd Hawaii International Conference on System Sciences, Kauai, HI, USA, 5–8 January 2010; pp. 1–10. [CrossRef]
22. Wita, R.; Jiamnapanon, N.; Teng-Amnuay, Y. An ontology for vulnerability lifecycle. In Proceedings of the 2010 Third International Symposium on Intelligent Information Technology and Security Informatics, Jinggangshan, China, 2–4 April 2010; pp. 553–557. [CrossRef]
23. Lee, Y.; Woo, S.; Song, Y.; Lee, J.; Lee, D.H. Practical Vulnerability-Information-Sharing Architecture for Automotive Security-Risk Analysis. *IEEE Access* **2020**, *8*, 120009–120018. [CrossRef]
24. Stellios, I.; Kotzanikolaou, P.; Grigoriadis, C. Assessing IoT enabled cyber-physical attack paths against critical system. *J. Comput. Secur.* **2021**, *107*, 102316. [CrossRef]
25. Rostami, S.; Kleszcz, A.; Dimanov, D.; Katos, V. A Machine Learning Approach to Dataset Imputation for Software Vulnerabilities. In Proceedings of the 10th International Conference on Multimedia Communications, Services and Security, Krakow, Poland, 8–9 October 2020; pp. 25–36. [CrossRef]
26. Sion, L.; Tuma, K.; Scandariato, R.; Yskout, K.; Joosen, W. Towards Automated Security Design Flaw Detection. In Proceedings of the 2019 34th IEEE/ACM International Conference on Automated Software Engineering Workshops, San Diego, CA, USA, 10–15 November 2019; pp. 49–56. [CrossRef]
27. Almorsy, M.; Grundy, J.; Ibrahim, A.S. Collaboration-based cloud computing security management framework. In Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing, Washington, DC, USA, 4–9 July 2011; pp. 364–371. [CrossRef]
28. Kotenko, I.; Doynikova, E. The CAPEC based generator of attack scenarios for network security evaluation. In Proceedings of the 2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Warsaw, Poland, 24–26 September 2015; pp. 436–441. [CrossRef]
29. Xianghui, Z.; Yong, P.; Zan, Z.; Yi, J.; Yuangang, Y. Research on parallel vulnerabilities discovery based on open source database and text mining. In Proceedings of the 2015 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Adelaide, Australia, 23–25 September 2015; pp. 327–332. [CrossRef]
30. Ruohonen, J.; Leppanen, V. Toward Validation of Textual Information Retrieval Techniques for Software Weaknesses. *Commun. Comput. Inf. Sci.* **2018**, *903*, 265–277. [CrossRef]
31. Guo, M.; Wang, J. An ontology-based approach to model common vulnerabilities and exposures in information security. In Proceedings of the ASEE Southest Section Conference, Marietta, GA, USA, 5–7 April 2009.
32. Shah, S.; Mehtre, B.M. An overview of vulnerability assessment and penetration testing techniques. *J. Comput. Virol. Hacking Tech.* **2015**, *11*, 27–49. [CrossRef]
33. Khera, Y.; Kumar, D.; Sujay, S.; Garg, N. Analysis and Impact of Vulnerability Assessment and Penetration Testing. In Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Prespectives and Prospects, Faridabad, India, 14–16 February 2019; pp. 525–530. [CrossRef]

34. Grigoriadis, C. Identification and Assessment of Security Attacks and Vulnerabilities, Utilizing CVE, CWE and CAPEC. Master's Thesis, University of Piraeus, Piraeus, Greece, 2019.
35. Scikit-Learn. Available online: https://scikit-learn.org/stable/ (accessed on 16 June 2021).
36. Tensorflow Hub. Available online: https://tfhub.dev/ (accessed on 16 June 2021).
37. Sentence Transformers Documentation. Available online: https://www.sbert.net/ (accessed on 16 June 2021).
38. Xia, P.; Zhang, L.; Li, F. Learning similarity with cosine similarity ensemble. *Inf. Sci.* **2015**, *307*, 39–52. [CrossRef]
39. Hafiz, M.; Adamczyk, P.; Johnson, R. Growing a pattern language (for security). In Proceedings of the ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software, Tucson, AZ, USA, 19–26 October 2012; pp. 139–158. [CrossRef]
40. Biswas, B.; Mukhopadhyay, A.; Gupta, G. "Leadership in Action: How Top Hackers Behave" A Big-Data Approach with Text-Mining and Sentiment Analysis. In Proceedings of the 51st Hawaii International Conference on System Sciences, Honolulu, HI, USA, 2–6 January 2018; pp. 1752–1761. [CrossRef]
41. Samtani, S.; Chinn, R.; Chen, H.; Nunamaker, J.F., Jr. Exploring emerging hacker assets and key hackers for proactive cyber threat intelligence. *J. Manag. Inf. Syst.* **2017**, *34*, 1023–1053. [CrossRef]
42. Xia, T.; Washizaki, H.; Fukazawa, Y.; Kaiya, H.; Ogata, S.; Fernandez, E.B.; Kato, T.; Kanuka, H.; Okubo, T.; Yoshioka, N.; et al. CSPM: Metamodel for Handling Security and Privacy Knowledge in Cloud Service Development. *J. Syst. Softw. Secur. Prot.* **2021**, *12*, 1–18. [CrossRef]