Tracing CAPEC Attack Patterns from CVE Vulnerability Information using Natural Language Processing Technique

Kenta Kanakogi
Waseda University
kanakogi-soft@fuji.waseda.jp

Hironori Washizaki
Waseda University
washizaki@waseda.jp

Yoshiaki Fukazawa
Waseda University
fukazawa@waseda.jp

Shinpei Ogata
Shinshu University
ogata@cs.shinshu-u.ac.jp

Takao Okubo
Institute of Information Security
okubo@iisec.ac.jp

Takehisa Kato
Hitachi,Ltd.
takehisa.kato.wx@hitachi.com

Hideyuki Kanuka
Hitachi,Ltd.
hideyuki.kanuka.dv@hitachi.com

Atsuo Hazeyama
Tokyo Gakugei University
hazeyama@u-gakugei.ac.jp

Nobukazu Yoshioka
National Institute of Informatics
nobukazu@nii.ac.jp

Abstract

To effectively respond to vulnerabilities, information must not only be collected efficiently and quickly but also the vulnerability and the attack techniques must be understood. 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 the common attributes and approaches employed by adversaries to exploit known weaknesses. Because the information in these two repositories is not directly related, identifying the related CAPEC attack information from the CVE vulnerability information is challenging. One proposed method traces some related CAPEC-ID from CVE-ID through Common Weakness Enumeration (CWE). However, it is not applicable to all patterns. Here, we propose a method to automatically trace the related CAPEC-IDs from CVE-ID using TF-IDF and Doc2Vec. Additionally, we experimentally confirm that TF-IDF is more accurate than Doc2vec.

1. Introduction

System administrators spend a lot of time dealing with vulnerabilities due in part to their sheer volume. To effectively respond and mitigate vulnerabilities, not only must vulnerability information be collected efficiently and quickly but also the vulnerability and the attack techniques utilizing the vulnerability must be understood. For example, when performing a penetration test, it is essential to refer to information about known vulnerabilities and attacks.

To collect such information, knowledge repositories on cyber-security 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.

CVE and CAPEC should both be used to implement an efficient penetration test. A vulnerability scanner can automatically detect CVE-IDs, but the CVE does not contain attack information. Therefore, we add attack information using the CAPEC attack patterns. Because CVE and CAPEC are not directly related, identifying the related CAPEC attack information from the CVE vulnerability information is difficult, especially for those without experience. 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 (Fig. 1). “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. CWE can trace the related CAPEC-ID from CVE-ID. In this paper, we refer to this method as the “conventional method”. The conventional method has two issues:

- It cannot trace some patterns of the related CAPEC-IDs from CVE-IDs when using CWE. Sections 3.2 and 5.3 mention specific patterns and conditions.
- Mappings between repositories are created

URI: https://hdl.handle.net/10125/71462
978-0-9981331-4-0

manually. Manual mapping cannot handle the growing amount of vulnerability information. In addition, the number of mapping failures is rising.

Currently, CVE and CAPEC are not explicitly mapped. It is preferable for the authors of CVE-ID to map it directly to CAPEC. However, accurate mapping is costly and difficult. This paper aims to trace the related CAPEC-ID directly from the CVE-ID. Our method suggests which CAPEC-ID should be related to a given CVE-ID. It does not provide a definitive "best" CAPEC-ID. Herein we calculate the similarity between CVE descriptions and CAPEC descriptions.

We propose two approaches to calculate similarity: Doc2Vec [4] and TF-IDF [5]. TF-IDF calculates the similarity of sentences by the number of occurrences of words, whereas Doc2Vec calculates the similarity of sentences by the distributed representation of sentences. We chose a representative similarity measure. These methods have been employed in previous studies [6], [7], but they have not been accurately evaluated or directly compared. Here, these two approaches are compared to the conventional one by tracing the related CAPEC-IDs from 44 CVE-IDs. Although other approaches can measure similarity such as LSI and Word Mover's Distance, evaluating these remains a future work. Our approach is described in detail in Section 4.

This paper aims to answer the following three Research Questions (RQs):

RQ1. How accurately can the relationships of security repositories be traced from CVE-ID to CAPEC-ID? This question assesses the trace accuracy of CAPEC-ID from CVE-ID.

RQ2. When using similarity based on natural language processing and machine learning, how accurate is the tracing from CVE-ID to CAPEC-ID? This question verifies the effectiveness of our proposed approach.

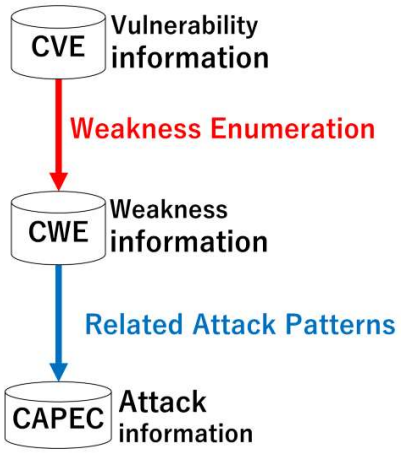

Figure 1. Relationships between security knowledge repositories

RQ3. Which of the three evaluated methods provides the best results? This question determines the most effective method among the conventional method and the two methods proposed in RQ2.

The contributions of this paper are twofold:

1. We clarify the mapping accuracy between security knowledge repositories.
2. Our method can easily identify CAPEC-IDs that are mapping candidates and assist in the mapping process.

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

2. Related Work

Previous studies have investigated vulnerabilities contained in repositories [6], [7], [8], [9], [10]. One study, which examined only the Software Defined Networking and Network Functions Virtualization vulnerability, also employed TF-IDF [6]. However, repositories have yet to be comprehensively evaluated. Although [7] had a similar objective as this study, they used Doc2vec without an explicit assessment, which made it difficult to evaluate the performance of the matching process.

Another study automatically mapped CVE to CAPEC and ATT&CK to find appropriate mitigation measures [8]. They created a neural network model with automatic classification in an attempt to realize a deep learning model that groups CWEs into CAPECs. However, they only suggested a method. On the other hand, this study conducts experiments with 44 CVE-IDs and prepares the correct CAPEC-ID.

Some studies examine the usage of topic modeling and natural language processing to extract hidden topics from the textual description of each attack pattern and learn the parameters of a topic model [11], [12]. 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 reports applications of topic analysis [13], [14]. In particular, one study proposes a hybrid method combining TF-IDF-weighted Doc2Vec and TF-IDF-weighted VSM [13]. This approach should be useful to improve the similarity measurement in this study.

Other studies investigated vulnerability ontology models [15], [16], [17]. They researched vulnerability models based on well-known public databases in the field of security such as CVE, CWE, and CAPEC.

Several studies have investigated security measures using security knowledge repositories. One study created a vulnerability management ontology that ranks attacks by security knowledge repositories [18]. Another study defined a framework to prioritize vulnerabilities [19]. Several studies have focused on mining methods and information retrieval for a security knowledge repository [20], [21], [22], [23], [24]. These papers mined each repository using their relationships. However, verifying the accuracy of the identified relationships were beyond the scope of these studies. Herein we evaluate the accuracy of the relationships between repositories.

Our method can propose attack patterns for a Penetration Tester. Several papers on penetration testing have introduced the Vulnerability Assessment and Penetration Test process [25], [26], [27]. Similar to this study, the authors of [27] performed penetration testing using CVE and CAPEC. Unlike our method, their method used the relationships between knowledge repositories.

3. Background

3.1. Security Knowledge repository

Many repositories disclose information about vulnerabilities. Here, a security knowledge repository is described.

Common Vulnerabilities and Exposures (CVE). CVE is a dictionary of common identifiers for known vulnerabilities. It includes more than 130,000 vulnerabilities.

National Vulnerability Database (NVD) [28]. NVD is the U.S. government repository of standards-based vulnerability management data, which is represented using the Security Content Automation Protocol (SCAP). NVD is fully synchronized with CVE. It includes the Common Vulnerability Scoring System (CVSS), Common Weakness Enumeration (CWE), Common Platform Enumeration (CPE), and other related database information.

Common Weakness Enumeration (CWE). CWE is a list of common software security weaknesses. It identifies categories of vulnerabilities. Each CWE-ID is assigned to create a hierarchical structure. Each CWE-ID is documented with a panoply of information, including a description, related Attack Pattern (CAPEC-ID), etc.

Common Attack Pattern Enumeration and Classification (CAPEC). CAPEC is a comprehensive dictionary of attack patterns employed by adversaries to exploit known weaknesses. Attack patterns are descriptions of common attributes and approaches used by attackers to exploit known weaknesses. CAPEC helps understand the specific elements of an attack and how to prevent a successful attack.

3.2. Motivating Example

Although CWE can trace some of the related CAPEC-ID from CVE-ID, it cannot trace all patterns. An example is CVE-2018-18442, which is a vulnerability related to a weakness due to flooding of network packets. 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. [29]

There is an attack pattern identifier for Flooding in CAPEC-125. However, CVE-2018-18442 is also related to CWE-20. Because CAPEC-125 cannot be traced from CWE-20, CAPEC-125 cannot be identified. By tracing the relationship between security knowledge repositories, we found that the correct CAPEC-ID could not be identified. The exact number of CVE-IDs that cannot be mapped to CAPEC via CWE is unknown. Since the issue is the use of CWE, it is preferable to directly trace from CVE to CAPEC. This is our motivation. Section 5.2 also references problems about CVE-IDs, which are impossible to map to CAPEC via CWE.

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

We considered how to trace CAPEC from CVE directly. We used similarity to trace the related CAPEC-IDs from a CVE-ID. Figure 2 overviews our method, which consists of three steps. First, a CVE-ID is inputted. For example, CVE-2018-18442 is used as the input data in Fig. 2. Second, the similarity between the description of the inputted CVE-ID and that of all CAPEC-IDs is calculated. Finally, the CAPEC documents are sorted by the similarity score.

We investigated two methods for the similarity measurements: TF-IDF and Doc2Vec. TF-IDF evaluates the importance of words in a document. It is a simple natural language process and is typically used to search for similar documents. On the other hand, Doc2Vec creates a paragraph vector and calculates the similarity. A machine learning model is built through averaging, combining, and estimating. Therefore, the results of the natural language processing depend on the

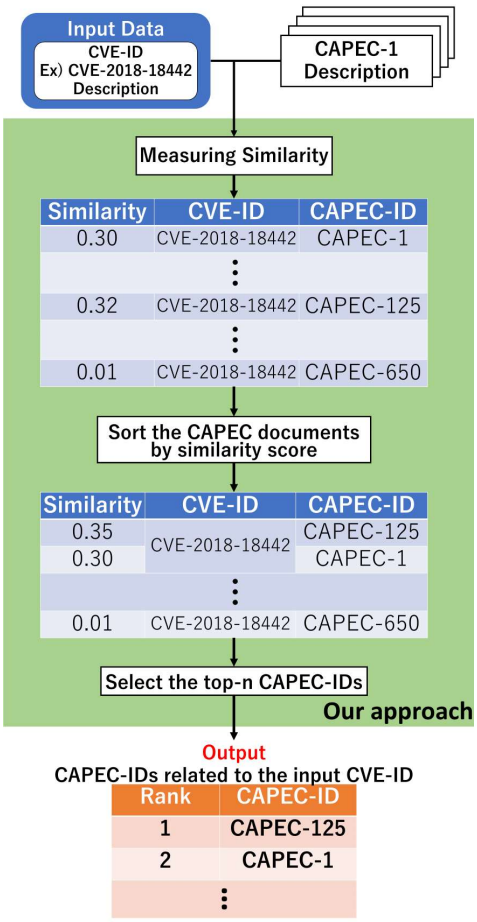

Figure 2. Overview of our method

machine learning model.

We describe each approach in detail below. Both approaches find the related CAPEC-ID in CVE-2018-18442. Eventually, we aim to identify CAPEC-125 from the CAPEC-ID.

4.1. Tracing method based on Doc2Vec

Doc2Vec uses the genism [30] doc2vec python library. The embedding vectors obtained from the text are used to calculate the similarity using the cosine similarity. A value close to 1 indicates similarity, whereas a value close to 0 indicates dissimilarity, which is typical for cosine trigonometric functions. Two models have been proposed to create a paragraph vector: The Distributed Memory Model of Paragraph Vectors (PV-DM) and the Distributed Bag of Words version of Paragraph Vector (PV-DBOW) Here, we use the PV-DBOW. DBOW ignores the word order, forcing the model to predict words randomly sampled from the paragraph in the output. Although PV-DM is more accurate because it considers the word order, we felt that the word order was not relevant when comparing the CVE and CAPEC descriptions. For example, the CVE describes it as a "brute force password attack", while CAPEC describes it as "brute force attack on passwords" or "Password Brute Forcing". Figure 3 shows the flow of the Doc2Vec approach using CVE-2018-18442 as an example. Table 1 shows the final results. The algorithm of the Doc2Vec approach is as follows:

STEP 1: Preprocess Data. The training data is the text of the Description section in CVE and CAPEC. All training data are given a Paragraph_id. There are 515 CAPEC-IDs (CAPEC-434 and 435 are excluded because their Description sections are blank). Each CAPEC-ID is given a Paragraph_id from 0 to 514. There are 131684 CVE-IDs. Each CVE-ID is given a Paragraph_id from 515 to 132198.

STEP 2: Create a Doc2Vec model. The formed training data is used to train the Doc2Vec model. Then a distributed representation of all the training data can be acquired.

STEP 3: Use the variance representation acquired in step 2 to find the similarity. Using the "self.wv.n_similarity" method [30] provides the similarity between paragraphs. The "self.wv.n_similarity" method gives two Paragpaph_id. This is the input data for our method. In Fig. 3, the Paragraph_id of CVE-2018-18442 is set to 1200. Then the Paragraph_ids, which range from 0 to 514, are inputted individually.

STEP 4: Obtain the similarity and sort. STEP 3 produces the similarity between the input Paragraph_ids. Consequently, the similarity scores between the inputted CVE-ID (CVE-2018-18442) and all CAPEC-IDs are determined. Then the scores are sorted in descending order. The ID with a higher rank is the related CAPEC-ID.

Table 1. List of CAPEC-ID rankings related to CVE-2018-18442 (Doc2Vec)

Rank	CAPEC-ID	Similarity
1	49	0.199
2	104	0.196
3	291	0.188
4	27	0.187
5	528	0.185
6	331	0.184
7	300	0.181
8	168	0.177
9	486	0.174
10	462	0.173
	⋮	
198	125	0.0659
	⋮	
515	503	-0.131

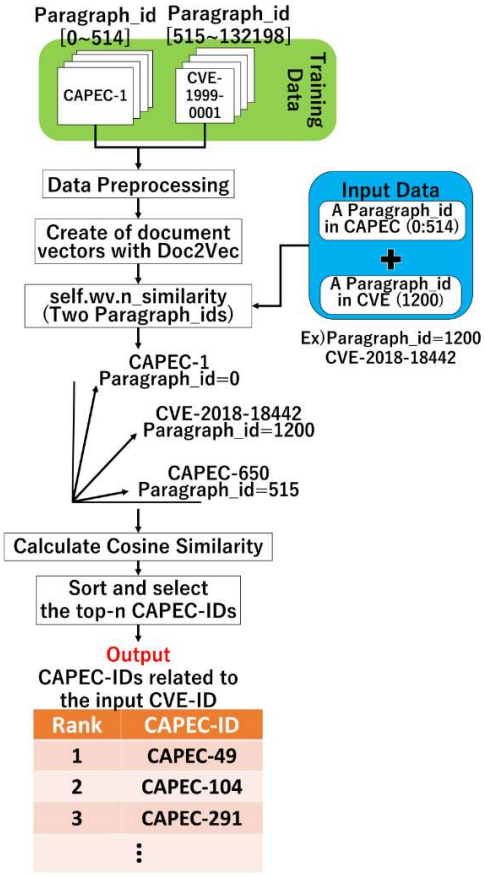

Figure 3. Overview of the tracing method based on Doc2Vec

4.2. Tracing method based on TF-IDF

In TF-IDF, we use scikit-learn [31]. Herein, we use the "TfidfVectorizer" method. Figure 4 shows the flow of the TF-IDF approach using CVE-2018-18442 as an example. Table 2 shows the final results. The algorithm of the TF-IDF approach is as follows:

STEP1: 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: Obtain a matrix of TF-IDF features. Convert a collection of corpuses to a matrix of TF-IDF features using "TfidfVectorizer".

STEP 4: Get the TF-IDF scores. Use "fit_transform" to learn the matrix of TF-IDF features and return the TF-IDF-weighted document-term matrix.

STEP 5: Sort the TF-IDF scores. STEP 4 produces the TF-IDF scores 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.

Table 2. List of CAPEC-ID rankings related to CVE-2018-18442 (TF-IDF)

Rank	CAPEC-ID	Similarity
1	49	0.199
2	104	0.196
3	291	0.188
4	147	0.140
5	488	0.129
6	184	0.122
7	469	0.121
8	594	0.116
9	125	0.100
10	308	0.979
	⋮	
515	650	0

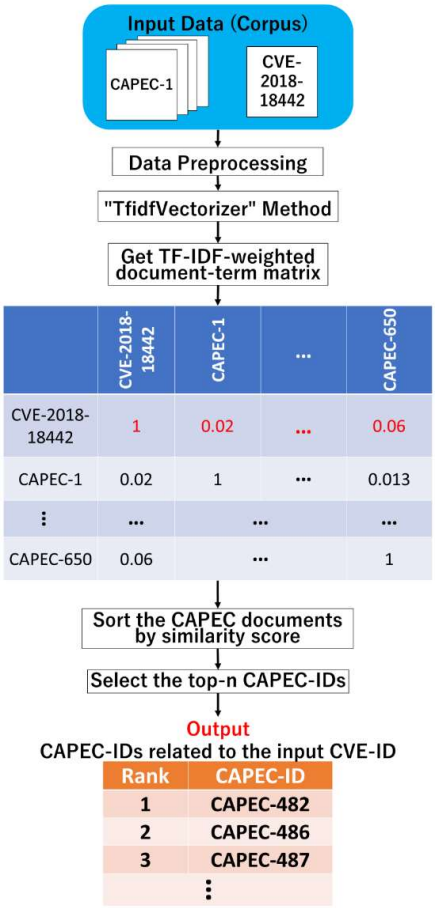

Figure 4. Overview of the tracing method based on TF-IDF

5. Experiments and Results

We prepared 44 CVE-IDs and used them as input data. We tested whether we could trace CAPEC-IDs related to each of the 44 CVE-IDs. In the evaluation, we

calculated the Recall@10. Recall@n indicates the proportion of relevant items found in the top-n recommendations. The RQs questions were answered using the tracing results of the 44 CVE-IDs.

5.1. 44CVE-IDs

CAPEC contains the Example Instance field, which lists the specific vulnerabilities targeted by this exploit instance of the attack. CVE-ID may be listed in this field. Example Instance field contained 7 CVE-IDs in 1999, 4 CVE-IDs in 2000, 4 CVE-IDs in 2001, 2 CVE-IDs in 2002, 1 CVE-ID in 2003, 3 CVE-IDs in 2004, 3 CVE-IDs in 2005, 13 CVE-IDs in 2006, 4 CVE-IDs in 2007, 2 CVE-IDs in 2010, and 1 CVE-ID in 2016. Hence, there are a total of 44 CVE-IDs listed. The average number of words is 36. The median is 34. The maximum is 81, and the minimum is 9. To evaluate the correctness of the tracing results using our method requires correct data, we selected these 44 CVE-IDs. If a CVE-ID listed in the Example Instance of a CAPEC-ID is used as input data, whether the corresponding CAPEC-ID is successfully traced can be verified. Originally, the mapping from CVE to CAPEC is many-to-many, but in this experiment, we assume that it is many-to-one.

5.2. RQ 1. How accurately can the relationships of security repositories be traced from CVE-ID to CAPEC-ID?

When tracing the relationships between repositories, we successfully traced 2 of the 44 CVE-IDs using the conventional method. This low accuracy is attributed to the relationship between CVE (NVD) and CWE. NVD is fully synchronized with CVE. We analyzed the accuracy of the mapping between CVE (NVD) and CWE.

The NVD webpage contains a section called “Weakness Enumeration”. This section provides information about the relationship between CVE-ID and CWE. There are four patterns of information in this relationship. Information in the first pattern is written with a CWE-ID. Information in the second pattern is written with multiple CWE-IDs. The information in the third pattern is written “NVD-CWE-Other”. Information in the fourth pattern is written with “NVD-CWE-noinfo”. In the cases of “NVD-CWE-Other” and “NVD-CWE-noinfo”, information cannot be traced to CAPEC-ID because it is not mapped to CWE-ID. Figure 5 shows the percentage of information for these relationships. Approximately 30% of CVE-IDs are not mapped to CWE-IDs.

When aggregated by year, the percentage of CVE-ID mapped to CWE-ID has increased each year (Fig. 6). In particular, the percentage of CVE-IDs mapped to CWE-ID has increased dramatically since 2008. The increasing percentage of CVE-IDs mapped to CWE-ID highlights the importance of accurate mapping of CWE-ID. However, bias is a problem. There are 839 CWE-IDs, of which only 149 are used to map CVE-IDs. Figure 7 shows the distribution for the top 15 mapped CWE-IDs listed.

Figure 7 focuses on CWE-20 and CWE-200. These two CWE-IDs have very high abstraction levels, indicating that many CAPEC-IDs are listed in the Related Attack Pattern section. CWE-20 has a relationship with 51 CAPEC-IDs, while CWE-200 has a relationship with 58 CAPEC-IDs. Due to these large numbers, it is difficult to identify which is the correct CAPEC-ID.

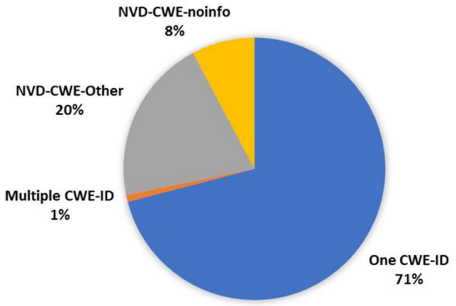

Figure 5. Percentage of CVE-IDs not mapped to CWE

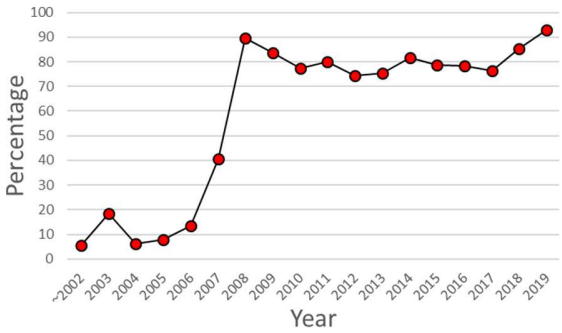

Figure 6. Changes in the percentage of CVE-ID that is mapped to CWE-ID

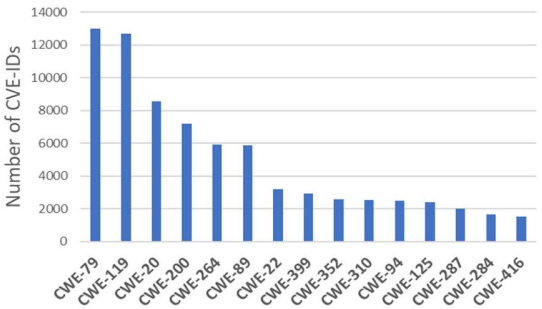

Figure 7. Vulnerability distribution by CWE-ID

We analyzed CWE-20 in detail. 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." [32]

As shown above, it is easy to understand why CWE-20 is often mapped from CVE-ID. However, CVE-IDs mapped for this reason cannot provide useful vulnerability information from CWE.

As discussed in Section 2.2, we introduced a pattern that CAPEC-ID cannot identify. CVE-2014-0160, which is a vulnerability about Buffer Overread, 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. [33]

CAPEC-540 contains an attack pattern identifier for Buffer Overread. There is a weakness identifier for Buffer Overread in CWE-126. However, the "Weakness Enumeration" section of CVE-2014-0160 says CWE-119 (Buffer Errors). Hence, CAPEC-540 cannot be traced from CWE-119. Why is it mapped to CWE-119? CVE-2014-0160 involves multiple weaknesses in CWE-125, 126, and 130. These three CWE-IDs are recognized as identifiers related to the Heartbleed bug. Despite this fact, we believe that the CWE-119 mapping 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.

Result of RQ 1. Only 2 out of the 44 CVE-IDs were traced to the related CAPEC-ID. Useful information about the attack that can be traced is inaccurate.

5.3. RQ 2. When using similarity based on natural language processing and machine learning, how accurate is the tracing from CVE-ID to CAPEC-ID?

The respective results of Doc2Vec and TF-IDF are described below. We traced the related CAPEC-IDs from the 44 CVE-IDs using the method described in Section 4.

Result of RQ 2. Doc2Vec traced 4 of the 44 CVE-IDs to the related CAPEC-ID. In contrast, TF-IDF traced 33 of the 44 CVE-IDs to the related CAPEC-ID.

5.4. RQ 3. Which of three evaluated methods provides the best results?

Figure 8 plots the experimental results for the Recall to trace related CAPEC-IDs from CVE-ID by the proposed method. TF-IDF yielded the best results. It successfully traced 33 of the 44 CVE-IDs. As a result, we could trace related CAPEC-IDs from CVE-ID in a keyword-based search. The CVE description word count does not affect the NLP approach. Currently, the search scope is narrow as it is limited to the text in the Description section of CAPEC. However, the accuracy of TF-IDF should improve upon widening the search scope. In addition, instead of relying on the TF-IDF measure, we would like to aggregate the TF-IDF measure and use the overlap score [34].

We believe that the corpus is a factor for why TF-IDF was more accurate than Doc2Vec. TF-IDF is better suited for smaller, more focused corpora. On the other hand, Doc2Vec is better for handling large corpora that span many topics. In this study, the accuracy of TF-IDF improved because the topic was limited to security. However, modifying the training data may enhance Doc2Vec.

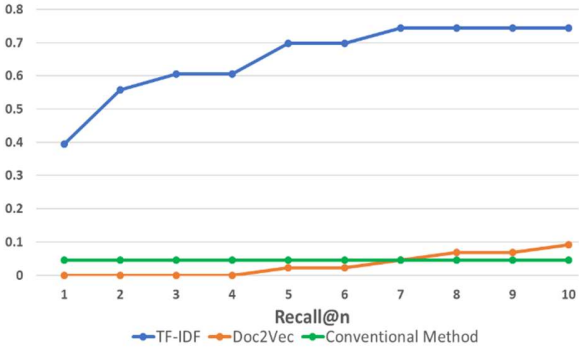

Figure 8. Recall for each approach

Herein all the text in the Description section of CVE and the Description section of CAPEC was trained as training data. The text in the Description section of CVE is often canned, which leads to biased training data. We believe that this bias decreased the accuracy. The reason for the low accuracy of the conventional method, as confirmed in RQ1, is the poor accuracy of the relationships between repositories. Most of the 44 CVE-IDs handled in the experiment were "NVD-CWE-Other". Hence, it cannot be traced to CAPEC-ID because it is not mapped to CWE-ID, resulting in a lower recall. Changing CVE-ID in the experiment should alter the result of the conventional method.

Result of RQ 3. All three methods can realize a trace, but the TF-IDF is the most accurate.

6. Conclusion

Herein we propose an approach to trace the related CAPEC-ID directly from 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 similarity. The similarity between CVE-ID and CAPEC-ID is calculated using two different measurements: Doc2Vec and TF-IDF. TF-IDF had a higher accuracy, but the results suggest that the Doc2Vec model can be improved.

Our method does not currently address the severity of each vulnerability. However, this is a topic for future work. Additionally, we need to enhance the accuracy of each approach. The first step is to expand the search scope to include text from other sections as well as that in the Description section. Moreover, the training data in Doc2Vec needs to be modified. The second step is to improve the accuracy of our training data using news articles about security as training data. Although we used CAPEC in this paper, other attack patterns are based on pattern language. In the future, we plan to evaluate attack patterns other than CAPEC as candidates for tracing. Moreover, we would like to improve the accuracy and increase the amount of information to provide useful cybersecurity information. 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 [35], [36] or extend the tracking by organizing the relationships between security concepts with metamodel [37].

7. Acknowledgement

The authors would like to thank the anonymous reviewers for their insightful comments and suggestions. This research was supported by the SCAT Research Grant; the MEXT enPiT-Pro Smart SE: Smart Systems and Services innovative professional Education program; the JSPS KAKENHI [grant number 16H02804]; and the JSPS KAKENHI [grant number 17K00475].

8. References

[1] Common Vulnerabilities and Exploits, https://cve.mitre.org/.

[2] Common Attack Pattern Enumeration and Classification, https://capec.mitre.org/.

[3] Common Weakness Enumeration, https://cwe.mitre.org/.

[4] Q. Le and T. Mikolov, "Distributed representations of sentences and documents", The 31st International Conference on Machine Learning, 2014, pp. 1188-1196.

[5] D. Miller, T. Leek, and R. Schwartz, "A hidden Markov model information retrieval system", The 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1999, pp. 214-221.

[6] Q. Dang and J. François, "Utilizing attack enumerations to study SDN/NFV vulnerabilities", The 4th IEEE Conference on Network Softwarization and Workshops, IEEE, Montreal, Canada, 2018, pp. 356-361.

[7] J. Navarro, V. Legrand, S. Lagraa, J. François, A. Lahmadi, G. D. Santis, O. Festor, N. Lammari, F. Hamdi, A. Deruyver, Q. Goux, M. Allard, and P. Parrend, "HuMa: A multi-layer framework for threat analysis in a heterogeneous log environment", In Foundations and Practice of Security - 10th International Symposium, Springer, 2017, pp. 144-159.

[8] E. Aghaei and E. Al-shaer, "ThreatZoom: Neural Network for Automated Vulnerability Mitigation", The 6th Annual Symposium on Hot Topics in the Science of Security, ACM, New York, USA, No.24, 2019, pp. 1-3.

[9] N. Scarabeo, B. C. Fung, and R. H. Khokhar, "Mining known attack patterns from security-related events", PeerJ Computer Science, vol. 1, Article number e25, 2015.

[10] X. Ma, E. Davoodi, L. Kosseim, and N. Scarabeo, "Semantic Mapping of Security Events to Known Attack Patterns", The 23rd International Conference on Applications of Natural Language to Information Systems, Springer, Paris, France, 2018, pp. 91-98.

[11] J. N. Ouchn, "Method and system for automated computer vulnerability tracking", Patent and Trademark Office, Washington, DC, U.S., U.S. Patent No. 9,871,815, 2018.

[12] S. Adams, B. Carter, C. H. Fleming, and P. A. Beling, "Selecting system specific cybersecurity attack patterns using topic modeling", The 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering, IEEE, New York, NY, USA, 2018, pp. 490-497.

[13] S. Ou and H. Kim, “Unsupervised Citation Sentence Identification Based on Similarity Measurement”, The 13th International Conference on Transforming Digital Worlds, Springer, Sheffield, UK, 2018, pp. 384-394.
[14] D. Kim, D. Seo, S. Cho, and P. Kang, “Multi-co-training for document classification using various document representations: TF–IDF, LDA, and Doc2Vec”, Journal of Information Sciences, Vol. 477, 2019, pp. 15-29.
[15] L. Zhu, Z. Zhang, G. Xia, and C. Jiang, “Research on Vulnerability Ontology Model”, The 2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference, IEEE, Chongqing, China, 2019, pp. 657-661.
[16] J. B. Gao, B. W. Zhang, X. H. Chen, and Z. Luo, "Ontology-based model of network and computer attacks for security assessment", Journal of Shanghai Jiaotong University (Science), vol. 18, no. 5, 2013, pp. 554-562.
[17] M. Ansarinia, S. A. Asghari, A. Souzani, and A. Ghaznavi, “Ontology-based modeling of DDoS attacks for attack plan detection”, The 6th International Symposium on Telecommunications, IEEE, Tehran, Iran, 2013, pp. 993-998.
[18] H. Wang, M. Guo, L. Zhou, and J. Camargo, “Ranking attacks based on vulnerability analysis”, The 43rd Hawaii International Conference on System Sciences, IEEE, Honolulu, HI, USA, 2010, pp. 1-10.
[19] R. Wita, N. Jiamnapanon, and Y. Teng-amnuay, “An ontology for vulnerability lifecycle”, The 2010 Third International Symposium on Intelligent Information Technology and Security Informatics, IEEE, Jinggangshan, China, 2010, pp. 553-557.
[20] M. Almorsy, J. Grundy, and A. Ibrahim, “Collaboration-based cloud computing security management framework”, The 2011 IEEE 4th International Conference on Cloud Computing, IEEE, Washington, DC, USA, 2011, pp. 364-371.
[21] I. Kotenko and E. Doynikova, “The CAPEC based generator of attack scenarios for network security evaluation”, The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IEEE, Warsaw, Poland, 2015, pp.436-441.
[22] Z. Xianghui, P. Yong, Z. Zan, J. Yi, and Y. Yuangang, “Research on parallel vulnerabilities discovery based on open source database and text mining”, The 2015 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IEEE, Adelaide, SA, Australia, 2015, pp.327-332.
[23] J. Ruohonen and V. Leppänen, “Toward Validation of Textual Information Retrieval Techniques for Software Weaknesses”, The 29th International Workshop on Database and Expert Systems Applications, Springer, Regensburg, Germany, 2018, pp. 265-277.
[24] M. Guo and J. A. Wang, “An ontology-based approach to model common vulnerabilities and exposures in information security”, ASEE Southest Section Conference, Semantic Scholar, 2009.
[25] S. Shah and B. M. Mehtre, “An overview of vulnerability assessment and penetration testing techniques”, Journal of Computer Virology and Hacking Techniques, Springer, vol. 11, no. 1, 2015, pp. 27-49.
[26] Y. Khera, D. Kumar, S. Sujay, and N. Garg, “Analysis and Impact of Vulnerability Assessment and Penetration Testing”, The International Conference on Machine Learning, Big Data, Cloud and Parallel Computing, IEEE, Faridabad, India, 2019, pp. 525-530.
[27] C. Grigoriadis, “Identification and Assessment of Security Attacks and Vulnerabilities, utilizing CVE, CWE and CAPEC”, University of Piraeus, Piraeus, Athens, Greece, 2019.
[28] National Institute of Standards and Technology. National Vulnerability Database, https://nvd.nist.gov/.
[29] The MITRE Corporation, “CVE-2018-18442”, available at https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-18442
[30] gensim: models.doc2vec – Doc2vec paragraph embeddings, https://radimrehurek.com/gensim/models/doc2vec.html.
[31] A. Swami and R. Jain, “Scikit-Learn: Machine Learning in Python”, Journal of Machine Learning Research, Vol. 12, 2011, 2825–2830.
[32] The MITRE Corporation, “CWE-20”, available at https://cwe.mitre.org/data/definitions/20.html
[33] The MITRE Corporation, “CVE-2015-0260”, available at https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160
[34] C. Manning, P. Raghavan, and H. Schütze, “Introduction to information retrieval”, Natural Language Engineering, 2010, pp. 100-103.
[35] B. Biswas, A. Mukhopadhyay, and G. Gupta, “"Leadership in Action: How Top Hackers Behave" A Big-Data Approach with Text-Mining and Sentiment Analysis”, The 51st Hawaii International Conference on System Sciences, Semantic Scholar, Honolulu, HI, USA, 2018, pp. 1752-1761.
[36] S. Samtani, R. Chinn, H. Chen, and J. F. Nunamaker Jr, “Exploring emerging hacker assets and key hackers for proactive cyber threat intelligence”, Journal of Management Information Systems, Semantic Scholar, 34(4), 2017, pp. 1023-1053.
[37] T. Xia, H. Washizaki, Y. Fukazawa, H. Kaiya, S. Ogata, E. B. Fernandez, T. Kato, H. Kanuka, T. Okubo, N. Yoshioka, and A. Hazeyama, “CSPM: Metamodel for Handling Security and Privacy Knowledge in Cloud Service Development”, Journal of Systems and Software Security and Protection, IGI-Global, 2021 pp.1-2.