HAL
archives-ouvertes.fr

Utilizing attack enumerations to study SDN/NFV vulnerabilities

Quang-Vinh Dang, Jérôme François

▶ **To cite this version:**

Quang-Vinh Dang, Jérôme François. Utilizing attack enumerations to study SDN/NFV vulnerabilities. IEEE ETSN - International Workshop on Emerging Trends in Softwarized Networks, Jun 2018, Montreal, Canada. hal-01763368v2

HAL Id: hal-01763368
https://hal.inria.fr/hal-01763368v2

Submitted on 6 Aug 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Utilizing attack enumerations to study SDN/NFV vulnerabilities

Quang-Vinh Dang
Inria Nancy Grand Est, France
quang-vinh.dang@inria.fr

Jérôme François
Inria Nancy Grand Est, France
jerome.francois@inria.fr

***Abstract*—Several cybersecurity attack enumerations area available today. These enumerations present lists of known attack patterns (CAPEC), security weaknesses (CWE) or cybersecurity vulnerabilities (CVE). These enumerations are being developed separately and manually.**

In this paper, we present the efforts in determining the relations between enumerations automatically. We rely on text-based, graph-based and recommendation-based approaches. Then we present of using the prediction in recommending related attacks to SDN/NFV security issues.

Experimental results showed that we can actually infer real relations. Furthermore, the results gave some insights into how the enumerations are created and linked, and some suggestions to improve the process in the future.

I. Introduction

Network softwarization with SDN and NFV attracts a lot of attention from academia and industry in recent years. Indeed, while the use of softwarized networks is increased, more vulnerabilities will be found and used by attackers. In order to handle efficiently security threats, security experts share their knowledge. Hence, most important and common attack patterns and vulnerabilities are collected and organized in attack enumerations. Recent years have seen attack descriptions related to softwarized networks.

Using standard knowledge representations and enumerations is a popular approach in dealing with security threats [1]. Several well-known security enumerations are developed and maintained that provide to the community a well-annotated knowledge repository on cyber-security issues. CAPEC (Common Attack Pattern Enumeration and Classification), CWE (Common Weakness Enumeration) and CVE (Common Vulnerabilities and Exposures) are among them [2]. The enumerations provide open-source lists of security issues that can be used by both researchers and practitioners.

Indeed, there are relations between these enumerations. By presenting relations between them, we can extend our knowledge regarding a particular problem. For instance, when a reader checks a particular CVE item, she can benefit by checking other related CAPEC or CWE items. Nevertheless the relations are being created manually that cannot cope with the large scale of these enumerations.

However, few research exists on finding the relations between these enumerations automatically, as well as suggesting possible related items. In this paper we present several approaches for relation inference. Then we apply these approaches to find attack patterns and software weaknesses that are related to SDN/NFV software vulnerabilities.

A. Enumerations

MITRE, which is a not-for-profit company[1], maintains three security enumerations that are investigated in this paper.

- **CAPEC** (Common Attack Pattern Enumeration and Classification) [3] is a public catalog of common attack patterns classified intuitively.
- **CWE** (Common Weakness Enumeration) [4] is a formal list of software weakness types.
- **CVE** (Common Vulnerabilities and Exposures) [5] is a list of common identifiers for publicly known cybersecurity vulnerabilities.

Ranking by abstraction level, we could say that CAPEC provides abstract information about attack pattern, CWE provides a list of practical weaknesses that attackers can exploit, and CVE provides the practical methods that attackers can use.

Definitely there are relations between the enumerations. For instance, MITRE defined that the CAPEC item with ID "1" (Accessing Functionality Not Properly Constrained by ACLs) is related to several CWE items with the following IDs: 285 (Improper Authorization), 732 (Incorrect Permission Assignment for Critical Resource) etc.

However these relations have been defined manually. It raises several problems:

- It is difficult to cope with large enumerations manually.
- It is difficult to define relationships through multiple enumerations. For instance, there is no direct CAPEC-CVE relations defined. It will be more difficult if we include more enumerations in the future.
- It is highly probable to miss a potential relation, or create a not-related relation.

In this paper, we address two problems:

- How to infer a relation between items that belong to different enumerations? In other words, how to reconstruct the links established by human experts? We refer the problem as *link prediction*.
- Given an item, how to recommend related items that belong to other enumerations? In other words, we want

[1] https://www.mitre.org

to find potential links that human experts might miss. We refer the problem as *link recommendation*.

As a result, we provide the security practitioners with more useful information to understand and handle a security incidents, i.e. indicating other threats which seems very similar by nature and for which she can thus apply a similar corrective action.

In this paper, we present the enumerations as a graph with nodes as items and edges as relations between them. Hence the goal is to analyze and predict missing links. By using graph-, text- and matrix-based features, we can effectively predict the links that are established manually and recommend new relations that did not exist in the data. We particularly apply our techniques to threats and vulnerabilities related to softwarized networks.

The paper is organized as follows. In Section II we present the related studies. We describe our methods in details in Section III. The experimental results are discussed in Section IV. We conclude our paper and draw some potential future works in Section V.

II. Related Work

Since its beginning in 1999 [1], CAPEC, CWE and CVE have been used as important enumerations in security analysis [6], attack modelling [7] or risk assessment [8].

To the best of our knowledge, there is not yet a comprehensive study on relations between different enumerations. In fact, several studies focused on analyzing a single enumeration and its internal dependencies, or analyzing established relations that are created manually by researchers.

Several research works focus only on CAPEC data. The author of [9] presented a work on taxonomy analysis and visualization of CAPEC. The study analyzed the relations within CAPEC items in different visualized approaches such as clustering or hierarchical analysis. However, the study does not analyze the CAPEC items in the relation with external enumerations like CVE. Furthermore, the study focuses on established relations rather than discovering the new ones. Related to [9], the authors of [2] analyzed the established relations to build an ontology rather than studying the missing links between enumerations.

Regarding CVE analysis there exist the works of [10], [11] or [12]. The authors analyzed the internal dependencies inside the CVE enumeration but do not extend the analysis to external knowledge repositories.

In other further analyses and usage of ontology based on open attack enumerations [13] there is no inference on finding new relations between CAPEC, CWE and CVE.

The authors of [14] presented a similar approach with us by using TF-IDF (Term Frequency - Inverse Document Frequency) to measure the similarity of CAPEC items and event-log. However, the authors do not utilize graph and matrix-based information. Furthermore, the authors focused on a different objective: they want to match a CAPEC item with an event-log rather than a CWE or CVE item.

Partly the work of [6] shared a similar goal with our study. However, similar to [14], the authors of [6] do not utilize the graph information. Furthermore, the authors used *word2vec* [15] that usually requires a large training dataset, and the result of *word2vec* is not clear and easy to interpret as the result of TF-IDF. On the other hand, akin to [14], the authors of [6] utilized an unsupervised learning without an explicit evaluation. Hence, it is difficult to evaluate the performance of the matching process. In our study we evaluate the recommendation by both manual and automated evaluations.

In the PhD thesis of [16] the author elaborated hybrid-recommendation techniques for defender counteractions. However, as we will show in the Section IV, the traditional recommendation techniques are not quite useful in our scenario due to the sparsity of the graph.

In this paper, we analyze the external relations over multiple enumerations. We aim to not only analyzing and reconstructing established relations but also discovering and recommending the new relations that did not exist yet.

III. Methodology

1) Graph-based approach: By representing enumerations and relations between them as a graph, we turn the problem into link-prediction problem in graphs [17]. The nodes of the graph are items in enumerations, and links between them are relations defined in the enumerations. We consider only non-directed graphs, because the relations are bi-directional.

Intuitively, if two nodes are more similar, it is more likely that they are related. Therefore we need to define a similarity function.

We leveraged different similarity-based link-prediction algorithms: s_1, s_2 and s_3. We use Γ_x to denote the list of neighbors of node x in the graph.

- *Adamic-Adar index* [18] is a similarity measure that calculates the similarity score between two nodes as:

$$s_1(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{log|\Gamma_z|} \tag{1}$$

- *Resource allocation* [19] is motivated by the resource allocation process in complex graphs.

$$s_2(x,y) = \sum_{z \in \Gamma_x \cap \Gamma_y} \frac{1}{|\Gamma_z|} \tag{2}$$

  Adamic-Adar and Resoure Allocation metrics perform well in social networks [17] due to the fact that human might actively seek to new connections. However, in an extremely sparse graph, the existing relations might not enough to verify the similarity of the nodes. Hence we use the third metric that is Preferential Attachment.
- *Preferential attachment* is motivated from the Barabaśi-Albert graph model [20]. The main idea of the model is "the rich get richer", i.e. if a node has a high connection degree, it is more likely that it will be connected in the future. More precisely, in Barabaśi-Albert graph model nodes are added one by one. When a new node is added,

it will connect with each existing nodes with a probability that is proportional to the number of links that the existing nodes already have.

Formally, suppose a new node x is added to the graph $G =< V, E >$, the probability that x will connect to the node i is:

$$p_i = \frac{|\Gamma_i|}{\sum_{k \in V} |\Gamma_k|} \tag{3}$$

The preferential attachment similarity score is calculated as:

$$s_3(x, y) = |\Gamma_x||\Gamma_y| \tag{4}$$

2) Text-based approach: Because items are text-based description, comparing the text would be helpful to detect those which are related to each other. Text similarity calculation is an active research trend today. Multiple approaches have been defined in last several decades. However, in this study we focus on using Term Frequency (TF) - Inverse Document Frequency (IDF) [21] technique due to several reasons:

- TF-IDF does not require a huge training dataset.
- Based on our observations, the relations between two items usually can be determined by keywords. For instance, if two items both mention some keywords like *buffer over-read* or *ACL misconfiguration*, it is likely that the two items are related. Using TF-IDF we can effectively extract keywords from contents of enumeration items.
- The items of the security enumerations are written in semi-formal language, i.e. they follow some specific writing styles for scientific articles. Therefore we do not need to worry about the semantic difference.

TF of a term t in a document d is calculated as $count(t, d)/|d|$ and IDF of a term t for a document d in a set of document D is calculate as $log[n/df(t)] + 1$. Here, we notate $count(t, d)$ as the number of occurrences of term t in the document d, and $df(t)$ is the number of documents that contains the term t. The similarity score is calculated by using $cosine$ distance function. Therefore, a keyword specific to a document is a word which is frequent in this document but very rare in other ones.

In order to apply TF-IDF in an efficient manner, considered terms for calculation are limited. Stop-words and terms with a frequency higher than 95% are removed since they are not representative and so will have a very IDF factor. Also, the terms with a frequency lower than 5% are also removed as they are not representative of the document (a low TF value in that case).

3) Recommendation-based approach: If we present the enumerations and the relations as a matrix, the task of relation prediction turns to be matrix completion [22]. We applied non-negative matrix factorization (NMF) technique [23] from recommendation systems to fill out the missing values.

In order to doing so, we present the relations between enumerations as a matrix. For instance, we can present CAPEC-CWE relations as a matrix D with rows are CAPEC items and columns are CWE items. The value of the cell $D(i, j)$ should present the relation from CAPEC item i to the CWE item j. The task of predicting a link between a CAPEC item i and a CWE item j becoming the task of filling value for the cell $D(k, l)$.

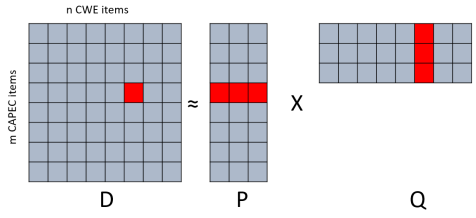

Fig. 1. Matrix factorization

The task is done by finding a factorization of a matrix D such as $D \approx P \times Q$ in such P presents CAPEC items and Q presents CWE items. The predicting value of $D(i, j) = P_i.Q_j^T$ is calculated by inner product. We visualize the process in Figure 1.

We note that all above methods are used for the link prediction task, but only text-based method is used for link recommendation task.

IV. Evaluation

A. Datasets

In this work, three public datasets are used:

- CAPEC[2] version 2.11 that contains 545 definitions.
- CWE[3] version 3.0 that contains 714 definitions.
- CVE[4] data created on 06-Feb-2018 that contains $122,866$ definitions.

We display the relations between three datasets in Figure 2. The links represent the relations between items in corresponding datasets. For instance, one CAPEC item can have relations with multiple CAPEC items as well as multiple CWE items.

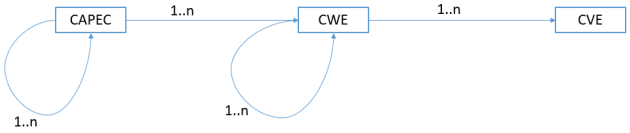

Fig. 2. Relations between datasets

B. Metrics

Because we address two problems (prediction and recommendation), two sets of metrics have been defined.

a) Term Definitions: To recall, if we represent the predicted labels and true labels as in Table I.

We calculate the accuracy, recall (or True Positive Rate), precision and FPR (False Positive Rate) as follows. Accuracy

[2]https://capec.mitre.org/data/
[3]https://cwe.mitre.org/data/index.html
[4]http://www.cve.mitre.org/data/downloads/index.html

	True labels	
Predicted labels	True Positive (TP)	False Positive (FP)
	False Negative (FN)	True Negative (TN)

TABLE I

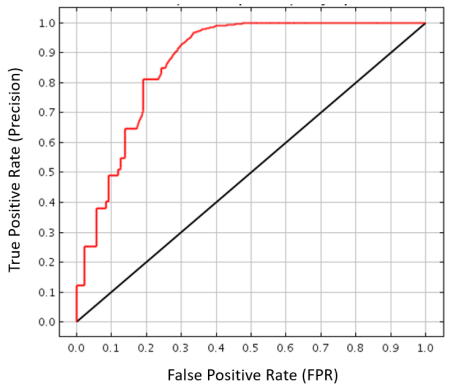

Fig. 3. Area Under Curve

measures the ratio of correct prediction over the total number of prediction. Precision measures the ratio of correct positive prediction over the total number of positive prediction. Recall measures the ratio of positive instances that the model can retrieve from the predictive dataset (testing set).

$$accuracy = TP + TN/(TP + FP + FN + TN) \quad (5)$$

$$recall = TP/(TP + FN) \quad (6)$$

$$precision = TP/(TP + FP) \quad (7)$$

$$FPR = FP/(FP + TN) \quad (8)$$

The AUC score is calculated as the area under the ROC curve as visualized in Figure 3. ROC curve is formed by two axes: TPR (i.e, precision) and FPR, i.e. it presents the performance of a model regarding a metric (TPR).

1) Link Prediction: We consider the problem of relation prediction as a classification, i.e. given two items, one is a CAPEC item and one is a CWE item, or one is a CWE item and one is a CVE item, we need to predict if there a relation between them exists or not. Therefore, the problem is a binary-prediction problem.

The datasets are actually severely imbalanced. Considering two datasets that contain M and N items respectively, the total number of possible relations between them is $M * N$. Hence, there is up to $389,130$ relations between CAPEC and CWE, and up to $87,726,324$ relations between CWE and CVE. By contrast there are only 700 CAPEC-CWE relations and $2,190$ CWE-CVE relations defined. Considering this issue, the regular *accuracy* metric should not be used [24] because it can be easily biased by majority classes, in that case the absence of a link.

Therefore, we used Area Under Curve (AUC) score to evaluate our prediction. AUC is suggested to be used in evaluating the prediction on imbalanced dataset [24] as [25] showed that is is a better metric in representing the performance of a learning algorithm compared to accuracy. We note that, given the output of our learning algorithm which can be considered as the probability of the existence of links, we prefer AUC to $F-1$ score. The reason is that AUC metric is defined on a list of probabilistic prediction while $F-1$ score is defined based on discrete predictions.

2) Link Recommendation: We select a set of SDN/NFV vulnerabilities from CVE dataset, apply our technique to recommend related attack patterns and security weaknesses and present them as a list of ranked related items ranking by the TF-IDF based similarity score. Then we assess manually the relatedness between the items.

In order to test the accuracy of the recommendation in a more automated manner, we also consider the metric *recall*. For instance, given a CVE item x. In the dataset, the human experts defined that x is related to CAPEC items y_1 and y_2. In recommendation phase when we temporarily remove the existing links connected to x and present a list of recommended CAPEC items that our system determined as related to x, these recommended links should also include y_1 and y_2.

C. Experimental Results

1) Link Prediction: In fact, the link prediction problem can be considered as the link-sign prediction problem in graphs [26] if we consider the existence of a link between two nodes as a link with value of "1" and the non-existence of a link is a link with value of "0".

We followed the evaluation configuration of [26]: to predict the link that already existed, we temporarily remove this link from the graph and use the remaining as the training data; and to predict the links that does not exist in the graph we simply use the entire original graph as the training data. The configuration is similar to "leave-one-out" cross validation [24]. We expect the predictive value in the former case will be close to 1 while the predictive value in the latter case will be close to 0. It is noteworthy that we expect that the values corresponding to the links that actually existed in the dataset should be close to "1". If there is a CAPEC item and a CVE item that seem to be related but the authors of the enumerations did not declare a relation between them, we expect that the value for this non-existed link should close to "0".

In fact, the graph features can be used directly as the unnormalized predictive values.

For instance, we can simply set a rule such as, if the graph feature is greater than 0.5 we will predict the missing link as an existed link and vice versa.

However, in order to increase the performance of the predictive algorithm we additionally performed logistic regression to train a classifier.

Input: Minority data $\mathfrak{D}^{(t)} = \{\boldsymbol{x}_i \in X\}$ where $i = 1,2,\dots,T$
Number of minority instances (T), SMOTE percentage (N), number of nearest neighbors (k)

for $i = 1,2,\dots,T$ **do**
1. Find the k nearest (minority class) neighbors of $\boldsymbol{x}_i$
2. $\widehat{N} = [N/100]$

while $\widehat{N} \neq 0$ **do**
1. Select one of the k nearest neighbors, call this $\overline{\boldsymbol{x}}$
2. Select a random number $\alpha \in [0,1]$
3. $\widehat{\boldsymbol{x}} = \boldsymbol{x}_i + \alpha(\overline{\boldsymbol{x}} - \boldsymbol{x}_i)$
4. Append $\widehat{\boldsymbol{x}}$ to $\mathcal{S}$
5. $\widehat{N} = \widehat{N} - 1$

end while
end for
Output: Return synthetic data $\mathcal{S}$

Fig. 4. SMOTE

We also applied sampling preprocessing techniques, i.e. over-sampling, under-sampling and SMOTE [27] to further improve the predictive power of the classifier.

Over-sampling means that we create multiple copies of the minority class while with under-sampling we only take a subset of majority classes into training dataset.

SMOTE [28] stands for *Synthetic Minority Over-sampling Technique*, means that we create synthetic training data to increase the number of instances from minority classes. The SMOTE algorithm works as follows.

Consider a dataset with two classes A and B with B is the minority classes. We aim to increase the number of instances of B. Suppose that we want to create new T instances of B. We select randomly T instances from the class B. For each instances, we find k nearest neighbours (from B only) and select randomly 1 neighbour from these k instances. The idea is that, the new created synthetic data point will lie somewhere in between the original data point and the neighbour. The pseudo-code is displayed in Figure 4.

First of all, we evaluated the performance of models using different graph features. The prediction result when we use only graph similarity values as a feature is shown in Figure 5. We observed that, using preferential attachment value achieves much higher AUC value than other similarity values. In fact, the best learning model achieves a very high AUC value of 0.925. From now on we use the term *graph feature* to refer to preferential attachment value.

Secondly, we verified the performance of learning models using each individual feature. We display the result in Figure 6. We observed that the model using graph feature individually has significant improvement compared to other features.

The results shed the lights on the development process of the enumerations. Theoretically, we might expect an uniform distribution of relations, i.e. for instance in linking a new CAPEC item to existing CWE items we should expect that each CWE item has a similar probability to be considered.

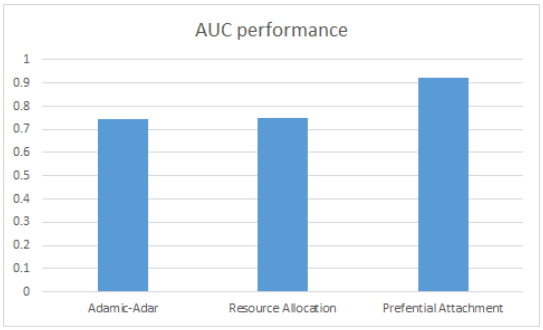

Fig. 5. Performance in term of AUC values using different graph similarity functions.

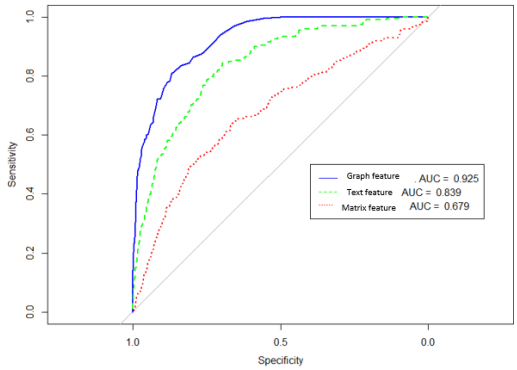

Fig. 6. Performance in term of AUC using different feature without sampling preprocessing.

However the results suggested that the development process of the enumerations follow the Barabaśi-Albert graph model, i.e. when a new item is created, the authors might follow the existing links to determine related items rather than scanning through the entire enumerations.

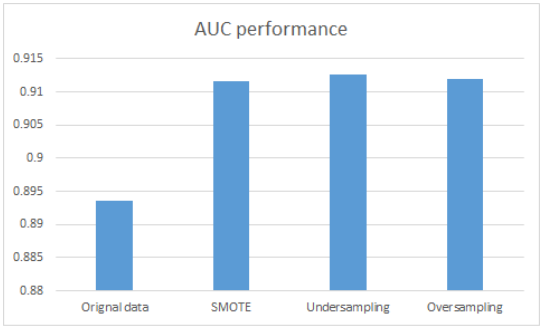

Fig. 7. Performance in term of AUC values using preferential attachment and TF-IDF values.

In order to validate the observation, we added text similarity feature into the learning model. We show the result in Figure 7. It turns out that adding text similarity feature decreases the performance of the learning model. In order to determine the links that will be actually added manually to the enumerations, graph features play a more important role than text similarity features.

Similarly, using recommendation-based approaches, i.e. matrix factorization, does not improve the performance of the learning model. It is mostly due to the cold-start problem where the matrix factorization techniques usually fail [29], i.e. it is because of the matrix is too sparse for the matrix factorization technique can be effectively performed.

2) Recommendation: We selected five CVE items that are related to SDN/NFV domains. The selected CVE items are CVE-2016-3708, CVE-2016-5363, CVE-2017-8189, CVE-2017-9265 and CVE-2017-1000411. To give the reader an idea of a CVE item, we present the content of existing CVE items as follows. We note that no relations from these CVE items to CAPEC or CWE were defined. Therefore, if any recommendation are proposed by our approach, it may bring new information to the enumerations.

> **CVE-2017-8189:** *FusionSphere OpenStack V100R006C00SPC102(NFV) has a path traversal vulnerability. Due to insufficient path validation, an attacker with high privilege may exploit this vulnerability to cover some files, causing services abnormal.*
>
> **CVE-2016-3708:** *Red Hat OpenShift Enterprise 3.2, when multi-tenant SDN is enabled and a build is run in a namespace that would normally be isolated from pods in other namespaces, allows remote authenticated users to access network resources on restricted pods via an s2i build with a builder image that (1) contains ONBUILD commands or (2) does not contain a tar binary.*

We calculated the similarity scores of these CVE items and recommend the top similar items from CAPEC and CWE enumerations.

For instance, regarding the item *CVE-2017-8189*, we recommend the following CAPEC and CWE items.

- **CAPEC-23**: *File Content Injection.*
- **CAPEC-17**: *Accessing, Modifying or Executing Executable Files.*
- **CAPEC-69**: *Target Programs with Elevated Privileges.*
- **CWE-22**: *Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal').*
- **CWE-99**: *Improper Control of Resource Identifiers ('Resource Injection').*
- **CWE-23**: *Relative Path Traversal.*

Regarding the item *CVE-2016-3708*, we recommend the following CAPEC and CWE items.

- **CAPEC-1**: *Accessing Functionality Not Properly Constrained by ACLs.*
- **CAPEC-88**: *OS Command Injection.*
- **CAPEC-498**: *Probe iOS Screenshots.*
- **CWE-285**: *Improper Authorization.*
- **CWE-749**: *Exposed Dangerous Method or Function.*
- **CWE-862**: *Missing Authorization.*

Regarding the subjective evaluation, i.e. based on manual evaluation, while a few recommendations might not be strong related ones, we can see that most recommendations are meaningful (such as CAPEC-17, 23 and CWE-22, 99 for CVE-2017-8189). Therefore, these recommendations can provide valuable information for the authors of the enumerations in the linking process.

Regarding the objective evaluation, we calculate *recall@10*, i.e. we use the top-10 related items as our recommendation. By this configuration, we achieved the $recall$ metric of 0.912. Hence we conclude that the recommendation are related to the given items.

V. Conclusions

In this paper, we research on the problem of automatic linking between different security enumerations and recommending attack patterns related to SDN/NFV vulnerabilities. The enumeration development follows the Barabasi and Albert model [20] and by using preferential attachment method we can effectively predict the links between enumeration items. On the other hand, by using TF-IDF method we can recommend missing links between enumerations. We demonstrated the usage of the method by recommending attack patterns and software weaknesses related to SDN/NFV vulnerabilities.

In the future, we might include more attack enumerations such as NVD[5] and FIRST[6] or real attack logs to analyze their relationship.

Acknowledgment

This work was partially funded by HuMa, a project funded by Bpifrance and Region Grand Est under the FUI 19 framework, and by the NATO Science for Peace and Security Programme under grant G5319 Threat Predict: From Global Social and Technical Big Data to Cyber Threat Forecast. It is also supported by the High Security Lab hosted at Inria Nancy Grand Est.

References

[1] R. A. Martin, "Making security measurable and manageable," in *Military Communications Conference, 2008. MILCOM 2008. IEEE*. IEEE, 2008, pp. 1–9.

[2] J. A. Wang and M. Guo, "Ovm: an ontology for vulnerability management," in *CSIIRW*, 2009.

[3] MITRE, "Common attack pattern enumeration and classification (CAPEC)," *https://capec.mitre.org*, 2018.

[4] ——, "Common weakness enumeration (CWE)," *https://cwe.mitre.org*, 2018.

[5] ——, "Common vulnerabilities and exposures (CVE)," *https://cve.mitre.org*, 2018.

[6] 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 *FPS*, ser. Lecture Notes in Computer Science, vol. 10723. Springer, 2017, pp. 144–159.

[7] I. V. Kotenko and E. Doynikova, "The CAPEC based generator of attack scenarios for network security evaluation," in *IDAACS*. IEEE, 2015, pp. 436–441.

[8] S. Madria and A. Sen, "Offline risk assessment of cloud service providers," *IEEE Cloud Computing*, vol. 2, no. 3, pp. 50–57, 2015.

[9] S. Noel, "Interactive visualization and text mining for the capec cyber attack catalog," in *Proceedings of the ACM Intelligent User Interfaces Workshop on Visual Text Analytics*, 2015.

[5] https://nvd.nist.gov/

[6] https://www.first.org/global/sigs/vrdx/vdb-catalog

[10] Z. Chen, Y. Zhang, and Z. Chen, "A categorization framework for common computer vulnerabilities and exposures," *Comput. J.*, vol. 53, no. 5, pp. 551–580, 2010.

[11] D. Toloudis, G. Spanos, and L. Angelis, "Associating the severity of vulnerabilities with their description," in *CAiSE Workshops*, ser. Lecture Notes in Business Information Processing, vol. 249. Springer, 2016, pp. 231–242.

[12] S. Neuhaus and T. Zimmermann, "Security trend analysis with CVE topic models," in *ISSRE*. IEEE Computer Society, 2010, pp. 111–120.

[13] J. A. Wang, H. Wang, M. Guo, L. Zhou, and J. Camargo, "Ranking attacks based on vulnerability analysis," in *HICSS*. IEEE Computer Society, 2010, pp. 1–10.

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

[15] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013.

[16] K. B. Lyons, "A recommender system in the cyber defense domain," AIR FORCE INSTITUTE OF TECHNOLOGY WRIGHT-PATTERSON AFB OH GRADUATE SCHOOL OF ENGINEERING AND MANAGEMENT, Tech. Rep., 2014.

[17] V. Martínez, F. Berzal, and J. C. C. Talavera, "A survey of link prediction in complex networks," *ACM Comput. Surv.*, vol. 49, no. 4, pp. 69:1–69:33, 2017.

[18] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social Networks*, vol. 25, no. 3, pp. 211–230, 2003.

[19] T. Zhou, L. Lü, and Y.-C. Zhang, "Predicting missing links via local information," *The European Physical Journal B*, vol. 71, no. 4, pp. 623–630, 2009.

[20] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[21] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[22] R. Pech, D. Hao, L. Pan, H. Cheng, and T. Zhou, "Link prediction via matrix completion," *EPL (Europhysics Letters)*, vol. 117, no. 3, p. 38002, 2017.

[23] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *NIPS*. MIT Press, 2000, pp. 556–562.

[24] N. Japkowicz and M. Shah, *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.

[25] J. Huang and C. X. Ling, "Using AUC and accuracy in evaluating learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 3, pp. 299–310, 2005.

[26] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg, "Predicting positive and negative links in online social networks," *WWW*, 2010.

[27] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Comput. Surv.*, vol. 49, no. 2, pp. 31:1–31:50, 2016.

[28] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[29] J. Tang, X. Hu, and H. Liu, "Social recommendation: a review," *Soc. Netw. Analys. Mining*, 2013.