INDEX

[[1]](#_[1]_A_Comp) [[2]](#_[2]_Vulner) [[3]](#_[3]_Topic_) [[4]](#_[4]_CPE_an) [[5]](#_[5] Securi) [[6]](#_[6]_Enabli) [[7]](#_[7]_Trend_)  [[8]](#_[8]_Graph_)

2022 IEEE 22nd International Conference on Software Quality, Reliability, and Security (QRS)

#### [1] ***A Comprehensive Analysis of NVD Concurrency Vulnerabilities***

[<https://ieeexplore.ieee.org/document/10062465>]

They found that with the prevalence of multi-core operating systems and the significant improvement of multi-threaded program performance, concurrency programming has become a mainstream of programming development and has consequently given rise to the emergence of concurrency vulnerabilities.

They claim that there are a large number of studies solving software security problems, but the researchers analyzing concurrency vulnerabilities are much fewer. Which gave them a reason to study and make a comprehensive analysis of concurrency vulnerabilities.

They extracted 839 concurrency vulnerabilities from Common Vulnerabilities and Exposures (CVE) and conducted a comprehensive analysis of the trend, classifications, causes, severity, and impact. They propose that -

1. From 1999 to 2021, the number of concurrency vulnerability disclosures showed an overall upward trend.
2. In the distribution of concurrency vulnerability, race conditions account for the largest proportion.
3. The overall severity of concurrency vulnerabilities is medium-risk.  
   The number of concurrency vulnerabilities that can be exploited for local access and network access is almost equal, and nearly half of the concurrency vulnerabilities [377/839] can be accessed remotely.
4. The access complexity of 571 concurrency vulnerabilities is medium, and the number of concurrency vulnerabilities with high or low access complexity is almost equal.

**They were four major findings out of 9 that they made:**

Findings 3.1 is a trend analysis. The upward trend from 1999 to 2021. This indicates the growing complexity of the software and the increasing prevalence of multi-core systems, which amplify the need for better detection, localization, and fixing.

Findings 3.2 is a classification-based finding on concurrency vulnerabilities; they claim that the majority of the concurrency vulnerabilities are related to race conditions, with CWE-362 accounting for the largest proportion. They have a critical nature, causing severe security issues, such as remote attacks on various services.

Findings 3.3: Severity of concurrency Vulnerabilities: Most concurrency vulnerabilites are of medium risk, with CVSS base scores between 4.0 and 6.9. There are also notable instances of high-risk, i.e., above 7.0.

Findings3.4: Most of the concurrency vulnerabilities are exploited through local access (53%), the next is remote network access (45%). Moreover, not all concurrency vulnerabilities exploited via remote network access are high-risk.

Practical Implications:

* Choosing the data extraction approach
* Manual screening based on CWE
* Filter data by combining keywords.
* Filter data according to the vulnerability description.  
   data cleaning

The two most common types of concurrency vulnerabilities that they stated were race condition and denial of service.

Inference made from Authentication: The majority of concurrency vulnerabilities (785 out of 839) can be exploited without any authentication, highlighting the ease of exploitation.

Increasing authentication instances can reduce the risk of exploiting concurrency vulnerabilities.

**Trends and Implications for Society**

* The number of concurrency vulnerabilities is increasing, necessitating enhanced developer awareness and measures to mitigate these vulnerabilities.
* Researchers should prioritize addressing denial of service, execution of arbitrary code, and privilege escalation resulting from concurrency vulnerabilities.

**Historical and ongoing research**:

Previous research has focused on various aspects of concurrency vulnerabilities, including detection methods and vulnerability trends.

This analysis builds on past work by providing a comprehensive study of the intrinsic characteristics of concurrency vulnerabilities in the CVE database.

Recommendations for Developers and Security Administrators:

Improve system security by addressing race conditions and enhancing authentication mechanisms.

Pay particular attention to medium-risk vulnerabilities and those that can be exploited remotely.

Implement measures to prevent denial of service, execution of arbitrary code, and privilege escalation.

Overall, this comprehensive study highlights the need for continued vigilance and improvement in handling concurrency vulnerabilities, especially given their potential impact on system security.

*Contributors*:

Lili Bo 1, Xing Meng 1, Xiaobing Sun 1, Jingli Xia 1, and Xiaoxue Wu 1

1 Yangzhou University, Yangzhou, Jiangsu, China

2Key Laboratory of Safety-Critical Software, Ministry of Industry and Information Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, China

lilibo@yzu.edu.cn, MX120200535@yzu.edu.cn, xbsun@yzu.edu.cn \*corresponding author

2021 Ninth International Conference on Advanced Cloud and Big Data (CBD)

#### ***[2] Vulnerability knowledge extraction method based on a joint extraction model***

[<https://ieeexplore.ieee.org/document/9816246>]

The paper discusses a joint neural network model that combines an attention mechanism and a bi-LSTM network to analyze and classify relationships in sentences from the CVE (Common Vulnerabilities and Exposures) dataset. The model uses reinforcement learning to enhance its ability to recognize and distinguish high-quality sentences and to update its strategic network based on the rewards obtained during training.

Entity and Relationship Annotation: Entities and relationships in CVE descriptions were manually annotated. The entities include the vulnerability attack environment, object, method, and result. Relationships were categorized as follows:

* Vulnerability attack environment
* Target of vulnerability attack
* Vulnerability attack method
* Vulnerability attack result

#### **Model**

* **Reinforcement Learning Model**: This model addresses the noise problem in the dataset by treating it as a reinforcement learning process.
* **Neural Network Model**: A bi-directional long short-term memory (bi-LSTM) network with a sentence-level attention mechanism is used. Word embedding and location embedding are applied to process entities, relationships, and text inputs. The model utilizes reinforcement learning to handle noise and improve performance.
* **Vector Representation**: Each sentence is transformed into a vector comprising position and word vectors. The word vectors are obtained using the word2vec embedding matrix, while the position vectors represent the relative distance between the current word and two entities in the sentence.
* *Bi-LSTM Layer*: This layer mines hidden information in sentences through forward and backward LSTM operations.
* **Attention Mechanism**: It analyzes the weight of the input sentence vector to identify important words or entities, enhancing the bi-LSTM layer's ability to process semantic information.

The proposed framework effectively extracts entities and classifies relationships in CVE descriptions. The model's performance was evaluated on a test set, demonstrating superior performance compared to traditional methods. The construction of the CVE vulnerability knowledge graph provides valuable insights for developers, enabling better analysis of attack sources, methods, and paths.

* **Dataset**: The model is evaluated on 30,000 samples from the CVE dataset, with 90% used for training and 10% for testing. The dataset includes four positive relation classes and one negative relation class, with over 20% noise data added to both the training and test sets.
* **Evaluation Metrics**: Precision (P), recall (R), and F1-scores (F1) are used to measure the model's performance.

The proposed framework offers an automated and effective solution for extracting and classifying entities and relationships in CVE descriptions. The resulting vulnerability security knowledge graph provides valuable insights for developers, enhancing their ability to analyze and mitigate security vulnerabilities.

* **Comparison of Models**: The joint neural network model (Union-LSTM) shows higher recall and F1 values compared to the original LSTM model, indicating better performance. The Union-LSTM model also performs better in the presence of noisy data.
* **P@N Comparison**: The Union-LSTM model outperforms the original models in terms of average accuracy, demonstrating significant improvement with larger data sets.
* **Effectiveness of Reinforcement Learning**: Combining reinforcement learning with the neural network model improves its performance, particularly in handling noisy data.

**Neo4j Graph Database**: The extracted entity pairs and predicted relationships are used to construct a knowledge graph, enriching the network security knowledge graph.

*Contributors*:

Zhen Liu1,2; XiaoQiang Di1,2,3; Wei Song1,2; WeiWu Ren1,2,()

1 Jilin Key Laboratory of Network and Information Security, Changchun University of Science and Technology, Changchun, China

2 School of Computer Science and Technology, Changchun University of Science and Technology, Changchun, China

3 Information Center, Changchun University of Science and Technology, Changchun, China

[renww@cust.edu.cn](http://renww@cust.edu.cn/)

2020 International Conference on Artificial Intelligence, Big Data, Computing, and Data Communication Systems (icABCD)

#### ***[3] Topic Modeling and Classification of Common Vulnerabilities and Exposures Database***

**[**[**https://ieeexplore.ieee.org/document/9183814**](https://ieeexplore.ieee.org/document/9183814)**]**

* They state that the paper focuses on using the Common Vulnerabilities and Exposures (CVE) repository for various cybersecurity purposes and analyzes CVE entries from 2009–2019.
* Methods: It mentions the use of Latent Dirichlet Allocation (LDA) Topic Modeling and keyword matching to categorize CVEs according to OWASP Top-10 risks.
* Results: The abstract indicates that 121,716 unique CVEs were analyzed, and the method developed facilitates automatic analysis, contributing to the creation of a vulnerability taxonomy.
* Previous Work: The introduction references previous work where manual mapping of CVE topics to OWASP risks was done, setting the stage for the automation aspect introduced in this paper.
* It outlines the use of LDA for topic modeling and the creation of rules based on OWASP's definitions and standards for keyword matching.
* Contribution: The section emphasizes the contribution of automating the analysis process, which can handle large datasets more efficiently than manual methods.

**Topic Modeling**: It describes topic modeling as an unsupervised machine learning technique useful for analyzing large text data sets, mentioning its application in various tasks.

* **LDA (Latent Dirichlet Allocation)**: The introduction explains LDA's role in identifying topics within documents and mapping those topics based on word distributions.
* **Data Processing and Preprocessing**: It briefly touches upon the preprocessing steps required to clean the data before applying LDA, such as removing stopwords, lemmatization, and tokenization.
* **Phases of Methodology**:
* **Phase 1** (Data Preprocessing): Focuses on text cleaning and preparation using various libraries and techniques.
* **Phase 2** (Building the Topic Model): Discusses the creation of the LDA model, visualizing topics, and identifying keywords and their importance.
* **Phase 3** (Rule-based Topic Classification): Explains the development of rules for topic classification based on expert knowledge and OWASP standards and the use of these rules to map topics to OWASP Top-10 risks automatically.

The study presents a method for classifying CVE entries into OWASP Top-10 vulnerability categories using latent Dirichlet Allocation (LDA) and keyword matching techniques. The comparison between manual and automated mapping is quantified using the percentage of tokens for each vulnerability type over five-year spans from 1999 to 2019.

* Consistency Between Manual and Automated Mapping:
* The automated method shows high consistency with manual mapping, as evidenced by low coefficients of variation (CV) across different vulnerability types and time spans. The CV values, being <= 1, indicate a low variance and close alignment between the two methods.
* The average CV values for different five-year spans are relatively low (0.27, 0.11, 0.05, 0.27, 0.02), demonstrating the robustness of the automated mapping approach.
* Percentage of Token Contribution:
* The distribution of percentage tokens across different OWASP vulnerabilities for various time periods indicates shifts in the prevalence of certain vulnerability types over time.
* For instance, the percentage of tokens contributing to 'A5:2017-Broken Access Control' increased over time, reflecting perhaps a growing focus or reporting on access control issues.

**Manual vs. automated method**

* Manual mapping provided a baseline to compare the effectiveness and accuracy of the automated method.
* The close alignment between manual and automated results (as indicated by low CV values) validates the reliability of the automated keyword matching technique for vulnerability classification.

The implications of this research are significant for cybersecurity practices, particularly in the automated analysis and categorization of vulnerabilities. By aligning CVE entries with OWASP Top-10 vulnerabilities, security practitioners can better prioritize and address critical security risks. The methodology presented provides a scalable solution for handling large volumes of vulnerability data, paving the way for more advanced and automated security analysis tools.

 Contributors:

MounikaVanamala Department of Computer Science, North Carolina A&T State University, Greensboro, NC, USA [vanamala.mounika7@gmail.com](mailto:vanamala.mounika7@gmail.com)

Xiaohong Yuan Department of Computer Science, North Carolina A&T State University, Greensboro, NC, USA [xhyuan@ncat.edu](http://xhyuan@ncat.edu)

Kaushik Roy Department of Computer Science, North Carolina A&T State University, Greensboro, NC, USA [kroy@ncat.edu](http://kroy@ncat.edu/)

The 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications 22–25 September 2021, Cracow, Poland 978-1-6654-2605-3/21/$31.00 ©2021 IEEE

#### ***[4] CPE and CVE-based Techniques for Software Security Risk Assessment***

[<https://ieeexplore.ieee.org/document/9660968>]

The paper addresses the challenge of automating the mapping of software product names in system logs to their corresponding entries in open vulnerability databases, particularly focusing on improving the accuracy of identifying known vulnerabilities. The need for this research arises from the difficulties in matching software names due to inconsistencies and errors, which affect the effectiveness of vulnerability databases like the National Vulnerability Database (NVD).

**Proposed Technique**:

**Mapping Algorithm**:

* The paper introduces a technique for automated mapping of software product names to Common Platform Enumeration (CPE) entries using the Ratcliff/Obershelp algorithm. This algorithm compares strings by identifying the longest common substrings, making it effective for detecting similarities even when there are minor variations in naming.
* The proposed algorithm involves a recursive comparison of name parts and uses Ratcliff/Obershelp for both the complete names and individual components (vendor, name, version) of the CPE entries.

Implementation:

* The technique was implemented in a Python tool that integrates CPE matching with vulnerability searching and risk assessment. This tool automates the process of mapping software names to CPE entries, identifying vulnerabilities, and assessing security risks based on Common Vulnerability Scoring System (CVSS) scores.

Results:

* The technique showed an average accuracy of 79% in identifying vulnerabilities when tested on Windows systems. The accuracy reflects the tool’s effectiveness in mapping and risk assessment, though there is room for improvement.
* The paper compares the proposed approach with previous methods, such as those using regular expressions and the Levenshtein algorithm. The Ratcliff/Obershelp algorithm was chosen for its superior performance in handling inconsistencies and errors in software names.
* Unlike prior methods that may require manual verification or have limited accuracy, the proposed technique automates the process with a relatively high accuracy rate.
* The authors plan to enhance the tool by analyzing the impact of incorrect mappings on the security risk scores and improving the accuracy of the results.
* There is also a focus on integrating machine learning techniques to further refine the mapping process and handle incorrect mappings more effectively.

The proposed technique offers a practical solution for automating the identification of vulnerabilities in information systems. By leveraging the Ratcliff/Obershelp algorithm, the research contributes to improving the accuracy of vulnerability detection and security risk assessment. Future research aims to enhance the technique's performance and integrate advanced methods for better handling of mapping inaccuracies.

 Contributors:

Roman Ushakov 1, Elena Doynikova 2, Evgenia Novikova 2, and Igor Kotenko 2

1 Saint Petersburg Electrotechnical University ”LETI”, 5 Prof. Popov Str., St. Petersburg, Russia,

romik75263@gmail.com

2 St. Petersburg Federal Research Center of the Russian Academy of Sciences (SPC RAS),

14ya Liniya V.O., St. Petersburg, Russia, {doynikova, novikova, ivkote}@comsec.spb.ru

2010 IEEE 21st International Symposium on Software Reliability Engineering

#### ***[5] Security Trend Analysis with CVE Topic Models***

[<https://ieeexplore.ieee.org/document/5635130/footnotes#footnotes>]

The study applies topic modeling techniques to the Common Vulnerabilities and Exposures (CVE) database to uncover prevalent vulnerability types and emerging trends. It utilizes Latent Dirichlet Allocation (LDA) on CVE descriptions to identify significant trends and topics.

The CVE database is a key resource for tracking security vulnerabilities. Traditional methods for analyzing CVE data include manual classifications and fixed systems like the Common Weakness Enumeration (CWE). This study aims to use unsupervised learning techniques, specifically topic modeling, to discover trends without the limitations of fixed classifications.

**Methodology**:

Data Gathering:

* They extracted CVE entries from the National Vulnerability Database (NVD) up to 2009.
* Focused on CVE descriptions and dates, using the OSVDB for dating CVEs due to issues with CVE’s own published-datetime field.

Corpus Preparation:

* They applied Porter stemming to normalize words.
* They removed common stop words to focus on significant terms.

Topic Modeling with LDA:

* Applied latent Dirichlet allocation to classify CVE descriptions into 40 topics.
* Used LDA to find prevalent topics and their changes over time.

Trend Analysis:

* Evaluated topic trends using yearly topic probabilities.
* Compared topic trends over different years to detect emerging issues.

Causes and Impacts:

* Explored the distinction between causes and impacts in CVE descriptions using LDA.

Alignment with CWEs:

* Compared LDA results with CWE classifications to assess alignment and accuracy.

* Identified trends such as the decline in PHP-related vulnerabilities, flattening of buffer overflows, and rising importance of SQL injection, XSS, and application servers.
* Discovered emerging trends, such as the increasing relevance of application servers.
* Growth of CVE Entries:
* They noted a peak in CVE entries in 2006, followed by a decline in subsequent years. The decreasing trend could be attributed to a backlog in entry creation.
* Precision and recall:
* Evaluated the effectiveness of LDA by comparing it with CWE classifications using precision and recall metrics. They found an alignment between LDA topics and CWE categories.

**Discussion**:

* Emerging Trends: The study highlights the rise of SQL injection and XSS vulnerabilities and the growing significance of application servers.
* Automated Analysis: The LDA-based approach provides a more flexible and scalable method for analyzing large volumes of vulnerability data compared to fixed classification systems.

Conclusion: The paper demonstrates that unsupervised learning techniques like LDA can effectively analyze vulnerability reports and identify trends, providing valuable insights into emerging security threats.

Future Work:

* Improving topic modeling techniques to refine trend detection.
* Expanding the analysis to include more recent CVE data.
* 28 topics were identified, covering a range of vulnerabilities from buffer overflows to SQL injection and cross-site scripting.
* Surprises: Topics for Linux kernel issues, Microsoft Office, and Microsoft Windows were found. Resource management issues, including denial-of-service attacks and memory leaks, were notably high.
* PHP Vulnerabilities: The frequency of PHP-specific vulnerabilities has declined since 2007.
* Declining Vulnerabilities: Buffer overflows, format string vulnerabilities, and resource management issues are decreasing.
* Surprises: SQL injection and cross-site scripting have shown some recent decline. The trend of arbitrary code execution vulnerabilities is also slightly down.
* Mapping Topics to CWEs: 12 of 24 topics mapped well to existing CWEs. Some discrepancies arise due to CWE assignments not fully aligning with the probabilistic nature of LDA topics.
* Challenges: Assigning multiple CWEs to a single CVE and differences in terminology can affect precision and recall.

 Contributors:

Stephan Neuhaus, Universit`a degli Studi di Trento, Trento, [Italy Stephan.Neuhaus@disi.unitn.it](mailto:ItalyStephan.Neuhaus@disi.unitn.it)

Thomas Zimmermann Microsoft Research, Redmond, WA, USA [tzimmer@microsoft.com](http://tzimmer@microsoft.com/)

2018 IEEE 18th International Working Conference on Source Code Analysis and Manipulation

#### ***[6] Enabling the Continuous Analysis of Security Vulnerabilities with VulData7***

[<https://ieeexplore.ieee.org/document/8530718>]

The paper introduces VulData7, a framework and dataset designed to facilitate the analysis and study of security vulnerabilities in open-source software. Here's a concise inference of the key points:

* VulData7 aims to address the challenge of collecting and analyzing real vulnerable code instances by providing a comprehensive dataset of security vulnerabilities.
* It focuses on four critical open-source systems: the Linux kernel, Wireshark, OpenSSL, and SystemD.
* The framework automates the extraction and linking of vulnerabilities reported in the National Vulnerability Database (NVD) with related code instances and patches.

**Dataset Coverage**:

* Includes vulnerabilities from 2,800 instances across the four systems.
* Provides detailed information such as CVE numbers, vulnerability descriptions, CWE classifications, CVSS severity scores, and associated patches.
* As of the latest update, it contains data for 1,600 fixed vulnerabilities out of 2,800 reported.
* Transformation and Extensibility:
* The framework is automated to continuously update and retrieve the latest vulnerability information.
* It extracts data from Git repositories and NVD feeds, making it flexible and easily extendable for additional projects.
* Usability:
* The dataset is available via a Java API and XML exports, providing utilities for easy access and analysis.
* It supports empirical studies, research, and educational purposes by providing a rich set of real-world vulnerability data.
* VulData7 is a valuable resource for developers and researchers focused on secure software development. It helps in understanding vulnerabilities and their fixes, which can be instrumental in improving security practices and tools.
* The framework's automation and extensibility make it a robust tool for ongoing research and education in software security.

In summary, VulData7 provides a comprehensive and automated solution for the collection and analysis of security vulnerabilities, offering valuable insights and tools for both researchers and developers.

**Enhancing Data Quality**:

* **Noise Reduction**: Future versions of VulData7 will aim to implement techniques to reduce noise and improve the quality of the dataset. This may involve more sophisticated filtering mechanisms to better distinguish between genuine vulnerabilities and unrelated changes.
* **Duplicate Handling**: To address duplicate instances due to commits in different branches, They aim to work on algorithms to identify and consolidate such cases, ensuring a cleaner dataset.
* **Additional SCM Systems**: Expanding support for other source code management systems like Subversion and Mercurial will increase the diversity and breadth of the dataset. This expansion is expected to include more software projects and contribute to a more comprehensive analysis.
* **Timeline Navigation**: Adding support for timeline navigation of commits and other advanced repository mining analysis tasks will provide more detailed insights into the history and context of vulnerabilities and fixes.
* **Integration with Existing Tools**: Enhancements to integrate VulData7 with existing static analysis tools, fuzzes, and other bug-finding techniques will support more effective evaluation and improvement of these tools.

VulData7 is designed as a robust framework , with its current capabilities, the tool provides valuable resources for researchers and practitioners aiming to understand and address software vulnerabilities.

* **Automation and Extensibility**: VulData7’s automated processes and extensible design make it a valuable tool for large-scale empirical research and analysis.

The framework is available under the Apache License (Version 2.0) and can be accessed on GitHub for those interested in contributing or utilizing the tool in their own research and development efforts.

Contributors:

Matthieu Jimenez Snt, University of Luxembourg [matthieu.jimenez@uni.lu](mailto:matthieu.jimenez@uni.lu)

Yves Le Traon Snt, University of Luxembourg [yves.letraon@uni.lu](mailto:yves.letraon@uni.lu)

Mike Papadakis, Snt., University of Luxembourg [michail.papadakis@uni.lu](mailto:michail.papadakis@uni.lu)

2011 IEEE International Conference on Privacy, Security, Risk, and Trust, and IEEE International Conference on Social Computing

#### ***[7] Trend Analysis of the CVE for Software Vulnerability Management***

[<https://ieeexplore.ieee.org/document/6113298>]

The research presented several insights into the trends of common vulnerabilities and exposures (CVEs) over a four-year period (2007–2010). Here are the key inferences drawn from the study:

1. **Overall Decrease in Vulnerabilities**: There was a significant 28% decrease in the overall frequency of vulnerabilities from 2007 to 2010. This decline may indicate improvements in secure coding practices or better vulnerability management processes. However, it is also possible that vulnerabilities discovered in 2010 could have been published later, suggesting a potential backlog.
2. **Severity Trends**: The proportion of high-severity vulnerabilities decreased over the period, with a noticeable shift towards medium- and low-severity vulnerabilities. Despite this, high-severity vulnerabilities still represented a substantial portion (over 45%) of the vulnerabilities reported in 2010.
3. **Exploitability** : A large majority (over 80%) of the vulnerabilities were found to be exploitable via network access without authentication. This highlights a critical area of concern for network security, as such vulnerabilities are more easily exploitable and can pose significant risks.
4. **CVSS Base Metrics**:

* *Access Complexity*: There was a notable decrease in vulnerabilities with low access complexity from 2009 to 2010, with a decline of 32%. This shift might reflect an increase in the complexity of attacks or improvements in mitigating lower-complexity vulnerabilities.
* *CIA Impacts*: Vulnerabilities with complete impacts on confidentiality, integrity, and availability (CIA) increased by approximately 9% over the study period. Conversely, vulnerabilities with partial CIA impacts decreased by around 12%. This indicates a shift towards vulnerabilities that affect all aspects of the CIA, potentially increasing their overall risk.

**Focus on Specific Vulnerability Types**: The research analyzed 15 specific vulnerability types that accounted for approximately 80% of all CVEs. For example, Denial of Service (DoS) vulnerabilities were particularly prevalent and showed an increasing trend in frequency and severity, especially in 2010.

1. Future Directions: The study suggests that further research should continue beyond 2010 to track emerging trends and vulnerabilities. This ongoing analysis can help IT professionals better understand evolving threats and improve security measures.

 Contributors:

Yung-Yu Chang, Pavol Zavarsky, Ron Ruhl, and Dale Lindskog

Information Systems Security Management

Concordia University, College of Alberta

Edmonton, Canada

yychang@csa.concordia.ab.ca, {ron.ruhl, pavol.zavarsky, dale.lindskog}@concordia.ab.ca

2022 IEEE International Conference on Big Data and Smart Computing (BigComp)

#### ***[8] Graph Summarization for Human-Understandable Visualization Towards CVE Data Analysis***

[<https://ieeexplore.ieee.org/document/9736492>]

The paper presents a method for summarizing large-scale CVE (Common Vulnerabilities and Exposure) data using graph-based techniques to aid security experts in understanding complex data more efficiently. Here’s a summary of the key inferences from the study:

**Graph Representation of CVE Data**:

* CVE data, which details software vulnerabilities, is represented as a graph where each CVE is a node. Nodes are connected if the similarity between their features is above a certain threshold, with the edge weight representing this similarity.

**Graph Summarization Techniques**:

* The study applies two graph embedding methods: HOPE (High-Order Proximity Preserved Embedding) and Laplacian Eigenmaps. These methods convert the graph into a latent vector space, preserving the structure and relationships in the original graph.

**Visualization and Clustering**:

* After embedding, the K-Means clustering algorithm is used to group nodes based on their embedded vectors. The results show that nodes with similar feature values are grouped together, making it easier to identify clusters of related vulnerabilities.
* The visualizations from HOPE show more distinct clusters compared to Laplacian Eigenmaps, indicating that HOPE might better preserve the cluster structures in the data.

**Practical Implications**:

* The summarized graph provides a more compact and interpretable view of the CVE data. By visualizing and clustering the graph, security experts can quickly identify and analyze patterns and relationships within the data, which can enhance their ability to respond to security threats.

**Potential Applications**:

* The graph summarization techniques used in this study could be beneficial not only for CVE data but also for other large-scale graph-based datasets in fields like bioinformatics, social networks, and defense systems, where understanding complex relationships is crucial.

In conclusion, the study effectively demonstrates that graph summarization and embedding techniques can simplify the analysis of large-scale data, making it more accessible and actionable for experts.

 Contributors:

Ji Sun Park Hanyang University, Seoul, South Korea [sgs04023@hanyang.ac.kr](http://sgs04023@hanyang.ac.kr/)

Mingu Kang Zhejiang University, Hangzhou, P.R. China [jiangmq@zju.edu.cn](http://jiangmq@zju.edu.cn/)

Sungryoul Lee The Affiliated Institute of ETRI Daejeon, South Korea, [srlee0525@nsr.re.kr](http://srlee0525@nsr.re.kr/)

Dong-Kyu Chae, Hanyang University, Seoul, South Korea [dongkyu@hanyang.ac.kr](http://dongkyu@hanyang.ac.kr/)