# Exploring the Propagation of Vulnerabilities from GitHub Repositories Hosted by Major Technology Organizations

**Ben Lazarine, Zhong Zhang, Agrim Sachdeva, Sagar Samtani, and Hongyi Zhu**

Indiana University and The University of Texas at San Antonio

**August 8, 2022**

# Introduction: Background & Motivation

- In recent year, GitHub has seen significant increase in organization users, with 4+ million organizations and 84% of the Fortune 100 companies utilizing the platform (GitHub, 2018).
  - Of the top 10 users with the most popular repositories, six are commercial technology organizations; Microsoft tops the list with over 21k forks and 128k stars (Analytics Vidhya, 2021).
- The scale and popularity of tech organization repositories poses significant potential security risks from the spread of vulnerabilities, and its implication has been mostly overlooked (Zhang et al, 2020).
  - E.g., security vulnerability CVE-2021-44228 in Log4j, which allowed attackers to execute malicious code on a system, had a broad impact since it is a popular library integrated in many applications.
- In this work, we aim to use graph embedding algorithms to identify the propagation patterns of vulnerabilities introduced by tech organization repositories in the GitHub network.

# Literature Review: Overview

- We review two areas of literatures to lay the foundations of this work:

1. **GitHub Vulnerability Assessment** to to identify prevailing methodologies and levels of analysis.

2. **Unsupervised graph embedding Algorithms** to identify methods for grouping repositories based on vulnerability similarity.

# Literature Review: GitHub Vuln. Assessment

| Year | Author | 1. Data Source | 2. Focus | Method | Level of Analysis |
|---|---|---|---|---|---|
| 2022 | Wartschinski et al. | 1,009 vulnerability-fixing commits from different GitHub repositories | Vulnerability Detection with Deep Learning on a Natural Codebase for Python | Recurrent Neural Network (LSTM) | Code Snippet |
| 2021 | Kaghazgaran et al. | 2,576 GitHub repositories (US 1,1451, China 1,125) | Measuring differences in country software repositories | Recurrent Neural Network (LSTM) | Country |
| 2021 | Qian et al. | 20,895 GitHub Repository | COVID-19 themed malicious repository detection | AHIN + AGCN + Meta – Learning | Repository |
| 2020 | Lazarine et al. | 258 GitHub repositories from NSF-funded Cyberinfrastructure group | Group key repository and users based on vulnerability | Unsupervised Graph Embedding | Organization & Repository |
| 2019 | Meli et al. | 681,784 GitHub Repository | Data leakage in public GitHub Repository | Regular Expression | API |
| 2018 | Kim et al. | 25,263 GitHub Repositories with C/C++ programs | Scalable detection of vulnerable code clones | VUDDY method: function-level granularity and a length-filtering technique | Code Snippet |

**Table 1. Selected Recent Studies Identifying Vulnerabilities in Source Code from GitHub.**

- **Key Observations and Research Gaps:**

1. Limited research has explored the influence of known vulnerable repository to other repositories on GitHub at the organizational level.
2. Past studies have primarily been focused on vulnerability detection including code snippet vulnerability detection, malicious repository detection, and key user identification.

# Literature Review: GitHub Vuln. Assessment

- Users and repositories on GitHub follow a bipartite graph structure for capturing relationships and nodal features (such as vulnerabilities, users) between repositories (Lazarine et al, 2020).

- To weight the connected edge (such as users) based on their shared attributes, an unsupervised feature weighting mechanism must be used during embedding generation.

- Prevailing unsupervised graph embedding methods that accounts for nodal features (e.g., Users) are summarized in Table 2.

# Literature Review: Unsupervised Graph Embedding Algorithms

| Category | Model | Projection Method | Nodal Features? | Enhance D-S Task? | Author | Year |
|---|---|---|---|---|---|---|
| Matrix Factorization | GF | Embedding inner products approximate edge weights between nodes | No | No | Ahmed et al | 2013 |
| | GraRep | integrates k-step relational information into learning process | No | No | Cao et al | 2015 |
| 1. | TADW | Formulate random walk as graph factorization along with node features | Yes | No | Yang et al | 2015 |
| | HOPE | Factorize high-order proximity matrix | No | No | Ou et al | 2016 |
| Random Walk | DeepWalk | Uniformed walk; Skip-gram | No | No | Perozzi et al | 2014 |
| | Node2vec | Biased-random walk; skip-gram | No | Yes | Grover et al | 2016 |
| | LINE | Combine 1st and 2nd order proximity feature extraction | No | No | Tang et al | 2015 |
| | HARP | Compress input graph to preserve higher-order structural features | No | Yes | Chen et al | 2018 |
| Deep Learning | SDNE | Autoencoder; reconstruct adjacency matrix | No | No | Wang et al | 2016 |
| | DNGR | Autoencoder; use probabilistic method to capture higher-order dependencies | Yes | Yes | Cao et al | 2016 |

**Table 2. Prevailing Unsupervised Graph Embedding Methods**

***Note: DNGR = Deep NN for Graph Representation; GF = Graph Factorization; GraRep = Graph Representations; HARP = Hierarchical Representation Learning; HOPE = High-Order Proximity-preserved Embedding; LINE = Large-scale Information Network Embedding; SDNE = Structural Deep Network Embedding.**

- **Key Observations:**

1. Two existing methods can account for nodal features: TADW, DNGR, and two existing methods have enhanced downstream tasks: Node2vec, HARP.
2. These selected methods can be adapted to analyze vulnerability spread between repos.

# Research Gaps and Questions

- Following research gaps were identified based on the literature review:
  1. limited research has explored the influence of known vulnerable repositories on other repositories on GitHub at the organization level.
  2. It is unclear how the vulnerabilities within these repositories propagate across GitHub due to forking or other sharing activity.
  3. How to leverage graph embedding-based techniques to represent a repository based on its vulnerabilities and relationships to other repositories requires additional investigation.
- Based on these research gaps, we pose the following research questions:
  1. What vulnerabilities exist in repositories hosted by major technology organizations?
  2. How do vulnerabilities propagate from an organization's repositories to its user's repositories?

# Research Design

- To answer these questions, we propose a research design that illustrated in Figure 1.

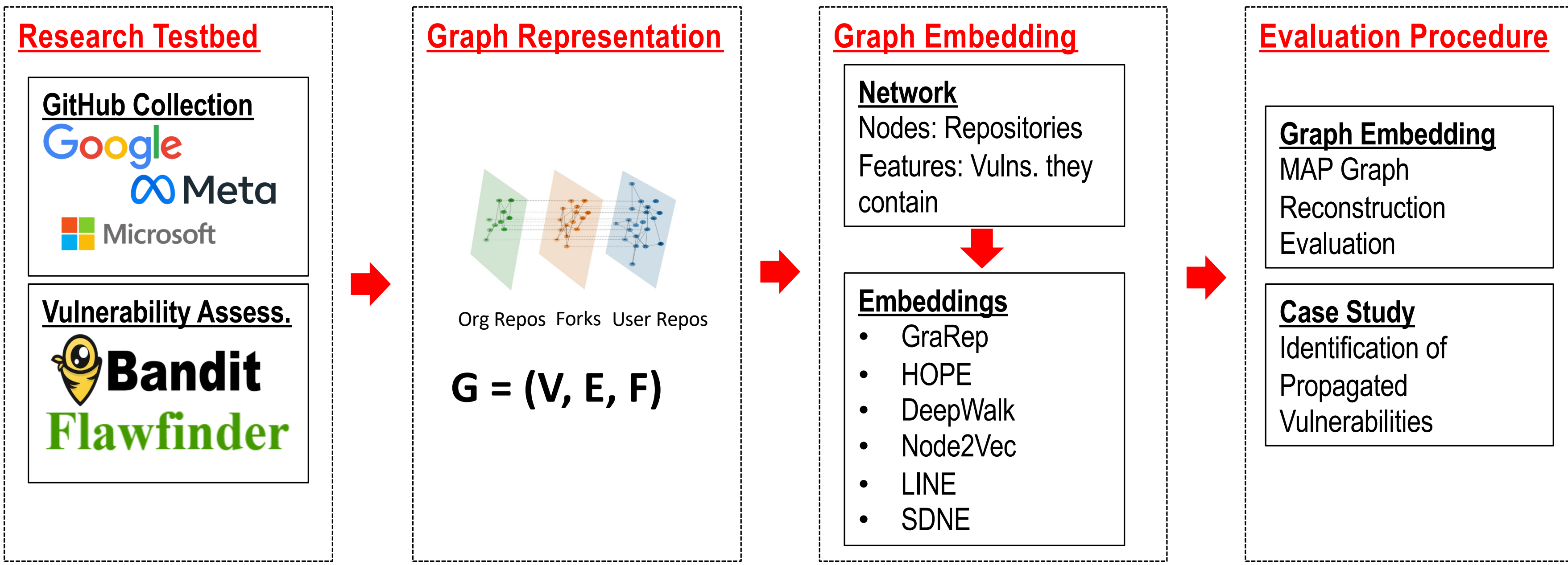



**Figure 1. Proposed Research Design.**

# Research Design:
# Data Collection & Vulnerability Assessment

| Company | Repositories | Seed Repositories | Forks | Vulnerabilities | |
|---|---|---|---|---|---|
| Meta | 504 | Facebook/Detectron | 504 | password | 383 |
| | | | | High Severity | 2,280 |
| | | | | Medium Severity | 4,220 |
| | | | | Low Severity | 3,598 |
| Google | 1,267 | Google/guava | 283 | password | 1,582 |
| | | | | High Severity | 8,491 |
| | | Google/styleguide | 480 | Medium Severity | 10,994 |
| | | Google-research/bert | 504 | Low Severity | 7,258 |
| Microsoft | 581 | Microsoft/TypeScript | 384 | password | 488 |
| | | | | High Severity | 50,872 |
| | | Microsoft/vscode | 197 | Medium Severity | 36,938 |
| | | | | Low Severity | 6,113 |

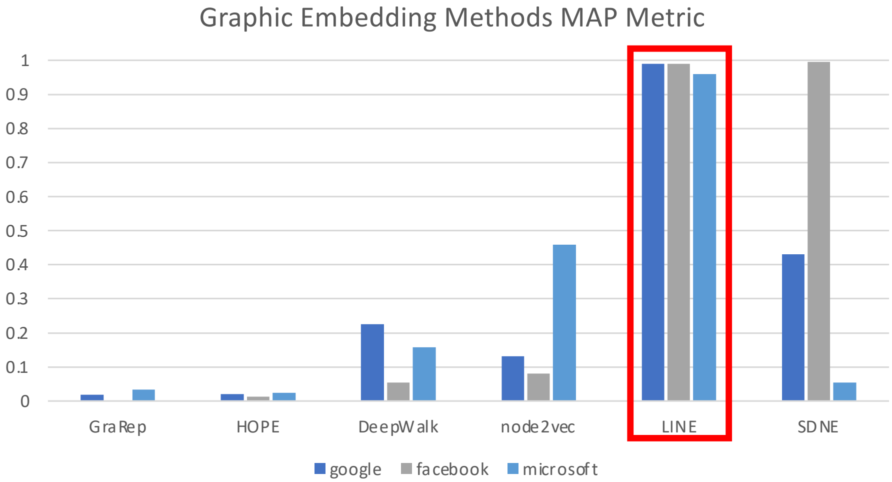
**Table 3. Summary of data collection and vulnerability assessment result.**

- **Key Observations:**
  - Overall, our vulnerability assessment identified 2,453 potential passwords, 61,643 high severity vulnerabilities, 52,152 medium severity vulnerabilities, and 16,969 low severity vulnerabilities.
  - High severity and medium severity vulnerabilities account for a large proportion of vulnerabilities across all three datasets.

# Research Design: Evaluation

- We evaluate six state-of-the-art unsupervised graph embedding models from three different categories:
- Matrix Factorization:
  - **GraRep**: symmetric and preserves high-proximity (Cao, 2015)
  - **HOPE**: directly models asymmetric similarities (Ou, 2016)
- Random Walk:
  - **DeepWalk**: learning latent representations of vertices in a network (Perozzi, 2014)
  - **Node2vec**: baseline for sequential methods which efficiently trade-offs between different proximity levels (Grover, 2016)
  - **LINE**: addresses the stochastic gradient descent limitation to improve efficiency on inference (Tang, 2015)
- Deep Learning :
  - **SDNE**: semi-supervised deep model to capture the highly non-linear network structure (Wang, 2016)

# Evaluation Result & Discussion




| Dataset | Embedding Method | | | | | |
|---|---|---|---|---|---|---|
| | **GraRep** | **HOPE** | **DeepWalk** | **Node2vec** | **LINE** | **SDNE** |
| Meta | 0.02 | 0.02 | 0.23 | 0.13 | 0.99 | 0.43 |
| Google | 0.00 | 0.01 | 0.05 | 0.08 | 0.99 | 0.99 |
| Microsoft | 0.03 | 0.02 | 0.16 | 0.46 | 0.96 | 0.05 |

**Table 5. Experiment Result in MAP (Mean Average Precision) Metric**

***Note: GraRep = Graph Representations; HOPE = High-Order Proximity-preserved Embedding; LINE = Large-scale Information Network Embedding; SDNE = Structural Deep Network Embedding.**

- **Key Observations:**
  - We found that random walk-based methods, specifically LINE, outperformed matrix factorization focused embedding methods with over 0.99 MAP for two of the dataset.
  - The Deep Learning method SDNE compared close to LINE with large datasets (for example, SDNE achieved 0.995 vs. LINE 0.990 with Google dataset)

# Evaluation Result & Discussion

- The spread of vulnerabilities through forking is seen in two seed repositories in our collection that had vulnerabilities identified: Google-research/Bert and Google/Styleguide.
  - 27 vulnerabilities were selected from Google-research/Bert, and in the nine forks of Bert sampled, 162 vulnerabilities were identified.
  - 16 vulnerabilities were selected from Google/Styleguide, and in the nine forks of Styleguide sampled, 288 vulnerabilities were identified.
- Additionally, Google-research/Bert and Google/Styleguide have 8.8 thousand and 12.2 thousand forks, respectively.
  - Forking quickly multiplies any vulnerabilities present in a seed repository, and vulnerabilities in fork instances can be preserved well after they have been addressed in the seed repository.

# Next Steps

- Our next steps are to conduct a case study that identifies how the vulnerabilities propagate.
- Future work could include formalizing “High Reputation” GitHub users and “Widely Forked Repositories.”
  - This comprehensive formalization can be a valuable extension of this work as it can provide the community with a clear definition of GitHub users and organizations that host repositories that may have significant vulnerability propagation.
- A second avenue for future research includes benchmarking vulnerability assessment scanners to ensure consistency of vulnerability assessment results and identify false positives.

# Reference


- Analytics Vidhya. (2021). Analyzing popular repositories on GitHub. [online] Available at: https://www.analyticsvidhya.com/blog/2021/07/analyzing-popular-repositories-on-github/#:~:text=Tensorflow%20is%20the%20most%2Dwatched%20and%20forked%20repository. [Accessed 11 Mar. 2022].
- Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V. and Smola, A.J., 2013, May. Distributed large-scale natural graph factorization. In Proceedings of the 22nd international conference on World Wide Web (pp. 37-48).
- Cao, S., Lu, W. and Xu, Q., 2015, October. Grarep: Learning graph representations with global structural information. In Proceedings of the 24th ACM international on conference on information and knowledge management (pp. 891-900).
- Cao, S., Lu, W. and Xu, Q., 2016, February. Deep neural networks for learning graph representations. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 30, No. 1).
- Chappell, B. (2019). NPR Choice page. [online] Npr.org. Available at: https://www.npr.org/2018/09/27/652119109/uber-pays-148-million-over-year-long-cover-up-of-data-breach.
- Chen, H., Perozzi, B., Hu, Y. and Skiena, S., 2018, April. Harp: Hierarchical representation learning for networks. In Proceedings of the AAAI conference on artificial intelligence (Vol. 32, No. 1).
- Faife, C. (2022). Wormhole cryptocurrency platform hacked for $325 million after error on GitHub. [online] The Verge. Available at: https://www.theverge.com/2022/2/3/22916111/wormhole-hack-github-error-325-million-theft-ethereum-solana.
- GitHub (2018). GitHub. [online] GitHub. Available at: https://github.com/.
- Grover, A. and Leskovec, J., 2016, August. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 855-864).
- Kaghazgaran, P., Lubold, N. and Morstatter, F., 2021. Organizational Artifacts of Code Development. arXiv preprint arXiv:2105.14637.
- Kim, S., Woo, S., Lee, H. and Oh, H., 2017, May. Vuddy: A scalable approach for vulnerable code clone discovery. In 2017 IEEE Symposium on Security and Privacy (SP) (pp. 595-614). IEEE.

# Reference


- Lazarine, B., Samtani, S., Patton, M., Zhu, H., Ullman, S., Ampel, B. and Chen, H., 2020, November. Identifying vulnerable GitHub repositories and users in scientific cyberinfrastructure: An unsupervised graph embedding approach. In 2020 IEEE International Conference on Intelligence and Security Informatics (ISI) (pp. 1-6). IEEE.
- Meli, M., McNiece, M.R. and Reaves, B., 2019, February. How Bad Can It Git? Characterizing Secret Leakage in Public GitHub Repositories. In NDSS.
- Ou, M., Cui, P., Pei, J., Zhang, Z. and Zhu, W., 2016, August. Asymmetric transitivity preserving graph embedding. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1105-1114).
- Perozzi, B., Al-Rfou, R. and Skiena, S., 2014, August. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 701-710).
- Qian, Y. and Zhang, Y., 2021, January. Adapting meta knowledge with heterogeneous information network for covid-19 themed malicious repository detection. In IJCAI.
- Wang, D., Cui, P. and Zhu, W., 2016, August. Structural deep network embedding. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1225-1234).
- Wartschinski, L., Noller, Y., Vogel, T., Kehrer, T. and Grunske, L., 2022. VUDENC: Vulnerability Detection with Deep Learning on a Natural Codebase for Python. Information and Software Technology, p.106809.
- Yang, C., Liu, Z., Zhao, D., Sun, M. and Chang, E., 2015, June. Network representation learning with rich text information. In Twenty-fourth international joint conference on artificial intelligence.
- Zhang, Y., Fan, Y., Hou, S., Ye, Y., Xiao, X., Li, P., Shi, C., Zhao, L. and Xu, S., 2020, August. Cyber-guided deep neural network for malicious repository detection in github. In 2020 IEEE International Conference on Knowledge Graph (ICKG) (pp. 458-465). IEEE.