Gregor Engels, Regina Hebig, Matthias Tichy (Hrsg.): SE 2023,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2023 125

VUDENC: Vulnerability Detection with Deep Learning on a Natural Codebase for Python – Summary

Laura Wartschinski[1] Yannic Noller[1] Thomas Vogel[1,2] Timo Kehrer[1,3] Lars Grunske[1]

Abstract: In this extended abstract, we summarize our work on VUDENC published in the journal Information and Software Technology (IST) in 2022 [Wa22]. VUDENC uses deep learning to learn features of vulnerable code from a real-world Python codebase and a network of long-short-term memory cells (LSTM) is then used to detect vulnerabilities in code at a fine-grained level.

Keywords: Vulnerability detection; Python; Deep learning

Context. Developing secure software system is important to mitigate vulnerabilities that otherwise could be exploited and have severe consequences such as loss, disclosure, or manipulation of data, or system failures. To avoid or at least reduce code flaws causing vulnerabilities, constructive engineering approaches can be used but need to be complemented by analyatical techniques to detect vulnerabilities. However, a manual detection of vulnerabilities requires expert knowledge and is time-consuming, and must therefore be supported by automated techniques.

Objective. We aim for an automated vulnerability detection technique that should achieve a high accuracy, point developers directly to vulnerable code fragments, scale to real-world software, generalize across the boundaries of a specific software project, and require no or only a moderate manual effort for the setup or configuration. We decided to focus on Python software, which has not been addressed yet by vulnerability detection with deep learning.

Method. To achieve these objectives, we developed VUDENC (Vulnerability Detection with Deep Learning on a Natural Codebase). It leverages deep learning to detect vulnerabilities in Python software. For this purpose, it automatically learns features of vulnerable code directly from a large and real-world Python codebase that we mined from GitHub and comprises (before filtering) 25,040 vulnerability-fixing commits in 14,686 different repositories. This dataset was labeled automatically according to the commit context and covers seven vulnerability types: SQL injection, Cross-site scripting (XSS), Command injection, Cross-site request forgery (XSRF), Remote Code Execution, Path disclosure, and Open Redirect. Accordingly, VUDENC can learn features for these types and detect vulnerabilities of these types. Using this dataset for training, VUDENC applies a word2vec model to identify semantically similar code tokens and provide a vector representation for deep learning.

[1] Humboldt-Universität zu Berlin, Department of Computer Science, Unter den Linden 6, 10099 Berlin, Germany.
{wartschinski, noller, thomas.vogel, kehrer, grunske}@informatik.hu-berlin.de
[2] University of Paderborn, Department of Computer Science, Warburger Straße 100, 33098 Paderborn, Germany.
[3] University of Bern, Department of Computer Science, Hochschulstrasse 6, 3012 Bern, Switzerland.

Particularly, a network of long-short-term memory cells (LSTM) is used to classify vulnerable code token sequences at a fine-grained level, highlight the specific areas in the source code that are likely to contain vulnerabilities, and provide confidence levels for its predictions.

Results. After determining suitable hyperparameters for the word2vec model and the LSTM model, we experimentally evaluated VUDENC on a test dataset of 1,009 vulnerability-fixing commits from different GitHub repositories and with different vulnerability types. VUDENC achieves a precision (i.e., the fraction of true positives in all positive predictions) of 82%-96%, indicating a very low false positive rate. The achieved recall (i.e., the rate of positives that were correctly identified in comparison to the total number of actual positives) of 78%–87% means that just 13%-22% of the samples labeled as vulnerable were missed. The overall F1 score (i.e., the harmonic mean of precision and recall) ranges from 80%-90%, which we see as a very satisfying result especially when considering related work and their effectiveness.

Conclusions. Our experimental results suggest that VUDENC is capable of outperforming most of its competitors in terms of vulnerably detection capabilities on real-world software. A comparable accuracy was only achieved on synthetic benchmarks, within single projects, or on a much coarser level of granularity such as entire source code files. In contrast, VUDENC uses a real-word dataset across multiple projects and detects vulnerabilities at the level of code fragments.

Data Availability. We provide a replication package covering the implementation and documentation of VUDENC on GitHub[4] and the following datasets on Zenode: the plain and embedded datasets for the seven types of vulnerabilities[5], the dataset of commits including their diff files mined from Github[6], and the Python corpus for training the word2vec model as well as one trained model[7].

Bibliography

[Wa22] Wartschinski, Laura; Noller, Yannic; Vogel, Thomas; Kehrer, Timo; Grunske, Lars: VUDENC: Vulnerability Detection with Deep Learning on a Natural Codebase for Python. Information and Software Technology, 144:106809, 2022.

[4] https://github.com/LauraWartschinski/VulnerabilityDetection

[5] https://doi.org/10.5281/zenodo.3559841

[6] https://doi.org/10.5281/zenodo.3559203

[7] https://doi.org/10.5281/zenodo.3559480