Modeling, classification, and detection of vulnerabilities and their variants in software code bases using AI

1. PhD title

Modeling, classification, and detection of vulnerabilities and their variants in software code bases using AI

2. PhD topic

Several databases of vulnerabilities exist. The purpose of this thesis is to model, classify, and generalise software vulnerabilities, based on existing vulnerability catalogues, in order to discover these vulnerabilities or their variants, or to find new related vulnerabilities, in code bases (public or private software code repositories).

3. PhD description

This thesis consists of setting up the appropriate modelling from a database (catalogue) classifying the vulnerabilities and making this modeling exploitable. Vulnerabilities could take various forms (anti-models of source code at the syntactic level, incorrect ASTs, use of deprecated APIs, etc.).

Specifically, the goal of the learning process will be to abstract and generalise vulnerabilities into siblings (a vulnerability and its variants).
This will enable robustness with regard to the code contexts in which vulnerabilities to be detected are inserted.

This last point is especially important because the design of contemporary computer systems is increasingly founded on the modularisation of these systems into a multitude of micro-services that offer fewer capabilities but interact intensively. This architecture is especially conducive to novel multi-stage attacks [13], which correspond to vulnerabilities that can be dispersed throughout the system.
Examples are syntactic source code antipatterns or the use of incorrect abstract syntax trees.
As a result of being embedded and dispersed throughout the remainder of the code, their detection is also more difficult.
Databases (or catalogues) that reference vulnerabilities do not always take this aspect into account, making it difficult to detect them efficiently.
Moreover, due to the heterogeneity of systems and microservices, it is essential to be able to generalise in order to account for potential variants or to anticipate the emergence of future variants.

The use of artificial intelligence techniques (specifically those based on machine learning) is a promising direction for analysing and defending against as many of these vulnerabilities (and their variants) as possible.
For these techniques to be effective, however, it is necessary to be able to abstract how these vulnerabilities manifest and for the selected abstraction to be capable of capturing their variants.

This work will rely as much as possible on public databases and existing developments, for example, MITRE CVE (Common Vulnerabilities and Exposures) [14], Software Heritage [15].

The expected outcome of this thesis is a system for modelling and detecting vulnerabilities and their variants in private or public code bases for civil and/or military applications.

4. PhD programme

This thesis will therefore first tackle the problem of finding a sufficiently robust abstraction to identify the vulnerabilities under consideration.
For this, the study will be based on existing abstractions (in particular those representing the code bases in the form of abstract syntax trees as well as those representing them in the form of graphs).

The selected abstraction will then be used to propose a classification of the considered vulnerabilities and its relevance will be evaluated on its capacity to take into account possible variants of these vulnerabilities.

All of this work will rely as much as possible on public databases and existing developments, for example, MITRE CVE (Common Vulnerabilities and Exposures) [14] as a vulnerability database, and Software Heritage [15] for software code repositories.

This thesis will take place within the DiverSE team of INRIA/IRISA/University of Rennes, under the responsibility of Olivier Barais (Professor at the University of Rennes, head of the DiverSE team), Paul Temple (Lecturer at the University of Rennes), and Olivier Zendra (Inria Research Fellow).

It will be carried out in connection with the team's other projects. In particular, the synergies identified with the Software Heritage Security - SWHSec project (in which DiverSE is co-leader), funded by the Cyber Campus [16], are as follows:

1. Reuse, in the context of this thesis, of the work done by an engineer in SWHSec to interface with existing catalogues of vulnerabilities (and their patches). This provides more input data for this thesis, without duplicating development that is already planned in SWHSec.

2. Use, in the context of this thesis, of Software Heritage as one of the (large) source code bases we can explore, benefiting additionally from the engineering work to facilitate access to SWH that is already planned in SWHSec WP1. Again, this provides more input for this thesis, without duplicating the development that is already planned in SWHSec.

3. Some of the research work done in SWHSec can be compared with the work done in this thesis. The latter is clearly more AI oriented than what is planned in SWHSec, and therefore complementary.

5. Applications

Applicants must have a Masters degree or equivalent, and originate from UE, UK or Switzerland.
To apply, please contact the advisors:
Olivier.Barais@irisa.fr, Paul.Temple@irisa.fr, Olivier.Zendra@inria.fr
(mailto:Olivier.Barais@irisa.fr,Paul.Temple@irisa.fr,Olivier.Zendra@inria.fr)

6. References

[1] Piergiorgio Ladisa, Henrik Plate, Matias Martinez, and Olivier Barais. “Taxonomy of Attacks on Open-Source Software Supply Chains”. In: Proceedings of the 44th IEEE Symposium on Security and Privacy, SP 2023, May 22-26, 2023, SAN FRANCISCO, CA. Ed. by IEEE Computer Society Technical Committee on Security and Privacy. IEEE, 2023, To appear. URI: https://doi.org/10.48550/arXiv.2204.04008

[2] Djamel Eddine Khelladi, Benoît Combemale, Mathieu Acher, Olivier Barais, and Jean-Marc Jézéquel. “Co-evolving code with evolving metamodels”. In: ICSE ’20: 42nd In- ternational Conference on Software Engineering, Seoul, South Korea, 27 June - 19 July, 2020. Ed. by Gregg Rothermel and Doo-Hwan Bae. ACM, 2020, pp. 1496–1508. doI: 10.1145/ 3377811.3380324. URl: https://doi.org/10.1145/3377811.3380324.

[3] Elliot Chikofsky and James Cross II. “Reverse Engineering and Design Recovery: A Taxonomy”. In: IEEE Software 7.1 (Jan. 1990), pp. 13–17. doI: 10.1109/52.43044. URl: http://dx.doi.org/10.1109/52.43044.

[4] European Commission. EU-FOSSA 2 - Free and Open Source Software Auditing. June 2020. URl: https://joinup.ec.europa.eu/collection/eu- fossa- 2/news/eu- fossa- 2- project-close (visited on 05/04/2022).

[5] Jean-Rémy Falleri, Floréal Morandat, Xavier Blanc, Matias Martinez, and Martin Mon- perrus. “Fine-grained and accurate source code differencing”. In: ACM/IEEE Interna- tional Conference on Automated Software Engineering, ASE ’14, Vasteras, Sweden - September 15 - 19, 2014. Ed. by Ivica Crnkovic, Marsha Chechik, and Paul Grünbacher. ACM, 2014, pp. 313–324. doI: 10.1145/2642937.2642982. URl: https://doi.org/10.1145/2642937. 2642982.

[5] White House. Readout of White House Meeting on Software Security. Jan. 2022. URl: https: //www.whitehouse.gov/briefing-room/statements-releases/2022/01/13/readout- of-white-house-meeting-on-software-security/ (visited on 05/04/2022).

[6] Hugo Martin, Mathieu Acher, Juliana Alves Pereira, et al. “Transfer Learning Across Variants and Versions: The Case of Linux Kernel Size”. In: IEEE Transactions on Software Engineering (2021), pp. 1–17. URl: https://hal.inria.fr/hal-03358817.

[7] Marc Ohm, Henrik Plate, Arnold Sykosch, and Michael Meier. “Backstabber’s Knife Collection: A Review of Open Source Software Supply Chain Attacks”. In: Detection of Intrusions and Malware, and Vulnerability Assessment. Ed. by Clémentine Maurice, Leyla Bilge, Gianluca Stringhini, and Nuno Neves. Cham: Springer International Publishing, 2020, pp. 23–43. IsBn: 978-3-030-52683-2.

[8] Marc Ohm, Henrik Plate, Arnold Sykosch, and Michael Meier. “Backstabber’s Knife Collection: A Review of Open Source Software Supply Chain Attacks”. In: Detection of Intrusions and Malware, and Vulnerability Assessment. Ed. by Clémentine Maurice, Leyla Bilge, Gianluca Stringhini, and Nuno Neves. Cham: Springer International Publishing, 2020, pp. 23–43. IsBn: 978-3-030-52683-2.

[9] OpenSSF. ossf/scorecard: Security Scorecards - Security health metrics for Open Source. Dec. 2020. URl: https://github.com/ossf/scorecard (visited on 05/04/2022).

[10] Henning Perl, Sergej Dechand, Matthew Smith, et al. “VCCFinder: Finding Potential Vulnerabilities in Open-Source Projects to Assist Code Audits”. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015. Ed. by Indrajit Ray, Ninghui Li, and Christopher Kruegel. ACM, 2015, pp. 426–437. doI: 10.1145/2810103.2813604. URl: https://doi.org/10.1145/ 2810103.2813604.![]

[11] Serena Elisa Ponta, Henrik Plate, and Antonino Sabetta. “Detection, assessment and mitigation of vulnerabilities in open source dependencies”. In: Empir. Softw. Eng. 25.5 (2020), pp. 3175–3215. doI: 10.1007/s10664-020-09830-x. URl: https://doi.org/10. 1007/s10664-020-09830-x.

[12] Marc Schönefeld. “Anti-patterns in JDK security and refactorings”. In: Detection of in- trusions and malware & vulnerability assessment, GI SIG SIDAR workshop, DIMVA 2004. Gesellschaft für Informatik eV. 2004.

[13] Fabio Pierrazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro. “Intriguing properties of adversarial ml attacks in the problem space”. In : 2020 IEEE symposium on security and privacy (SP). IEEE, 2020. p. 1332-1349.

[14] MITRE CVE: https://cve.mitre.org/index.html

[15] Software Heritage: https://www.softwareheritage.org/

[16] Campus Cyber: https://campuscyber.fr/

Last update: 14/04/2023 14:04