Gregor Engels, Regina Hebig, Matthias Tichy (Hrsg.): SE 2023,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2023 105

Introducing FUM: A Framework for API Usage Constraint and Misuse Classification

Michael Schlichtig[1], Steffen Sassalla[2], Krishna Narasimhan[3], Eric Bodden[4]

Abstract: Application Programming Interfaces (APIs) are the primary mechanism developers use to obtain access to third-party algorithms and services. Unfortunately, APIs can be misused, which can have catastrophic consequences, especially if the APIs provide security-critical functionalities like cryptography. Understanding what API misuses are, and how they are caused, is important to prevent them, e.g., with API misuse detectors. However, definitions for API misuses and related terms in literature vary. This paper presents a systematic literature review to clarify these terms and introduces FUM, a novel Framework for API Usage constraint and Misuse classification. The literature review revealed that API misuses are violations of API usage constraints. To address this, we provide unified definitions and use them to derive FUM. To assess the extent to which FUM aids in determining and guiding the improvement of an API misuses detector's capabilities, we performed a case study on the state-of the-art misuse detection tool CogniCrypt. The study showed that FUM can be used to properly assess CogniCrypt's capabilities, identify weaknesses and assist in deriving mitigations and improvements.

Keywords: API misuses; API usage constraints; classification framework; API misuse detection; static analysis

1 Classification of API Usage Constraints and Misuses with *FUM*

Reuse of functionalities and algorithms is key to software development and is enabled by APIs. However, misusing an API can cause serious consequences, such as program crashes or data leakage. We performed a systematic literature review on API misuses from November 2020 to February 2021 to improve our understanding of API misuses. We considered 69 publications to derive definitions and our classification **F**ramework for API **U**sage constraints and **M**isuses (*FUM*) [Sc22].

FUM is mainly based on the works of Monperrus et al. [Mo12], Amann et al. [Am19] and Li's refinement [Li20] on the work of Amann et al. [Am19]. API misuses are caused by the violation of an API usage constraint which is a restriction imposed by the API designer or expert to the API but cannot be checked by the programming language's compiler. Therefore,

[1] Heinz Nixdorf Institute, Paderborn University, Germany michael.schlichtig@uni-paderborn.de
[2] Hasso Plattner Institute, University of Potsdam, Germany steffen.sassalla@student.hpi.de
[3] Technische Universität Darmstadt, Germany kri.nara@cs.tu-darmstadt.de
[4] Heinz Nixdorf Institute, Paderborn University & Fraunhofer IEM Paderborn, Germany eric.bodden@uni-paderborn.de

FUM defines API usage constraint types and localizes them with the related part(s) of an API method call, e.g., a *Post-Call* is located at the *return value* (cf. Figure 1).

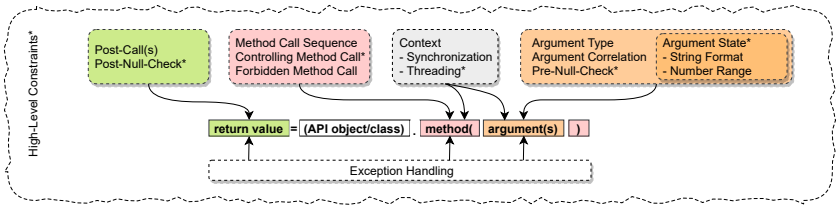

Fig. 1: *FUM* [Sc22] - Overview of API usage constraint types associated with parts of an API method call. Types marked with an asterisk are additions to the work of Monperrus et al. [Mo12]. Dashed colored boxes are specific to one single API method call part. Uncolored dashed boxes are API usage constraint types spanning multiple parts of an API method call.

2 Data Availability - Case Study

FUM was tested in a case study to evaluate a state-of-the-art cryptographic API misuse detector [Kr20] . The data of the case study is available at `https://doi.org/10.6084/m9.figshare.16832749` and contains the evaluation protocols of each API misuse sample and detailed theoretical explanations of suggested improvements.

Acknowledgements

Funded by the Deutsche Forschungsgemeinschaft (DFG) – SFB 1119 – 236615297

Bibliography

[Am19] Amann, S.; Nguyen, H. A.; Nadi, S.; Nguyen, T. N.; Mezini, M.: A Systematic Evaluation of Static API-Misuse Detectors. IEEE Transactions on Software Engineering, 45(12):1170–1188, 2019.

[Kr20] Krüger, Stefan : CogniCrypt - The Secure Integration of Cryptographic Software. Ph.D. thesis, University Paderborn, October 2020.

[Li20] Li, Xia: An Integrated Approach for Automated Software Debugging via Machine Learning and Big Code Mining. Ph.D. thesis, The University of Texas at Dellas, 2020.

[Mo12] Monperrus, Martin; Eichberg, Michael; Tekes, Elif; Mezini, Mira: What should developers be aware of? An empirical study on the directives of API documentation. Empirical Software Engineering, 17(6):703–737, Dec 2012.

[Sc22] Schlichtig, Michael; Sassalla, Steffen; Narasimhan, Krishna; Bodden, Eric: FUM - A Framework for API Usage constraint and Misuse Classification. In: 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). pp. 673–684, 2022.