# Program Synthesis for Binary-code Deobfuscation

Keywords: Binary analysis, deobfuscation, artificial intelligence, program synthesis

*Main Supervisor*
Grégoire MENGUY
*CEA LIST*
*gregoire.menguy@cea.fr*

*Senior Supervisor*
Sébastien BARDIN
*CEA LIST*
*sebastien.bardin@cea.fr*

Obfuscation [3, 4] aims to protect software from reverse engineering. It translates a program $P$ into a functionally equivalent program $P_o$, harder to analyze. While obfuscation is used to protect *Intellectual Property* and other valuable software assets, it is also used to protect malware. Thus, automated *deobfuscation* methods [15, 1, 5, 11, 14] have been proposed to cope with the quick advances in obfuscation. Given an obfuscated program $P_o$, the goal is to simplify it into a simpler yet functionally equivalent program $P^*$ – ideally, $P^*$ should be as simple as the original unprotected code $P$.

A wide variety of deobfuscation methods have been proposed. In particular, the deobfuscation of mixed-boolean arithmetic (MBA) expressions [16] is a hot topic in both academia [11, 1, 10, 9] and industry [7, 13, 12, 6]. Interestingly, the state-of-the-art considers that such MBA expressions are standalone and simplifies them one by one, with no contextual information. However, in practice, diversification methods are used [3, 4, 2] in obfuscated malware and goodware. Hence, multiple versions of the same obfuscated code is often available.

In this project, we aim to study how the access to multiple obfuscated versions of the same expression can help deobfuscation. We will especially focus on its impact on black-box deobfuscation [1, 11] based on program synthesis [8], an AI field aiming to generate code from input-output examples.

The project will proceed as follows:

- Get familiar with the state-of-the-art of black-box deobfuscation and MBA obfuscation;
- Create a dataset of MBA obfuscated expressions using the Tigress and OLLVM obfuscators, with multiple versions of each expression and evaluate Xyntia (the open source black-box deobfuscator from the CEA — `https://github.com/binsec/xyntia`) over it;
- Propose an extension in Xyntia to use the multiple versions of the expressions and compare this extension with Xyntia.

## 1 Expected deliverable

- A summary of the bibliography made by the student;
- A documented implementation of the proposed extensions;
- A final report (slides) with the first two deliverables and a summary of the research and results.

## 2 Organization

Regular meetings will be organized with the supervisor (online or in person at CEA Nano-Innov, Saclay).

## 3 How to apply

To apply, students must send an email to the main supervisor (Grégoire Menguy) with the senior supervisor (Sébastien Bardin) in copy.

# References


[1] Tim Blazytko et al. "Syntia: Synthesizing the Semantics of Obfuscated Code". In: USENIX *Security*. 2017.

[2] C. Collberg et al. *The Tigress C Diversifier/Obfuscator*. URL: http://tigress.cs.arizona.edu/.

[3] Christian Collberg and Jasvir Nagra. *Surreptitious Software: Obfuscation, Watermarking, and Tamperproofing for Software Protection*. Surreptitious Software: Obfuscation, Watermarking, and Tamperproofing for Software Protection, 2009.

[4] Christian Collberg, Clark Thomborson, and Douglas Low. *A taxonomy of obfuscating transformations*. 1997.

[5] Robin David, Luigi Coniglio, and Mariano Ceccato. "QSynth-A Program Synthesis based Approach for Binary Code Deobfuscation". In: *BAR 2020 Workshop*. Internet Society, 2020.

[6] Ninon Eyrolles, Louis Goubin, and Marion Videau. "Defeating MBA-based Obfuscation". In: *Proceedings of the 2016 ACM Workshop on Software PROtection, SPRO@CCS 2016, Vienna, Austria, October 24-28, 2016*. 2016.

[7] UNH SoftSec Group. *MBA-Solver Code and Dataset. https://github.com/softsec-unh/MBA-Solver*. 2021.

[8] Sumit Gulwani, Oleksandr Polozov, Rishabh Singh, et al. "Program synthesis". In: *Foundations and Trends® in Programming Languages* (2017).

[9] Jaehyung Lee and Woosuk Lee. "Simplifying Mixed Boolean-Arithmetic Obfuscation by Program Synthesis and Term Rewriting". In: *Conference on Computer and Communications Security*. 2023.

[10] Binbin Liu, Junfu Shen, and Jiang Ming et al. "MBA-Blast: Unveiling and Simplifying Mixed Boolean-Arithmetic Obfuscation". In: *USENIX Security*. 2021.

[11] Grégoire Menguy, Sébastien Bardin, and Bonichon et al. "Search-Based Local Black-Box Deobfuscation: Understand, Improve and Mitigate". In: *Conference on Computer and Communications Security*. 2021.

[12] Alex Petrov. *Hands-Free Binary Deobfuscation with gooMBA*. https://hex-rays.com/blog/deobfuscation-with-goomba. 2023.

[13] Benjamin Reichenwallner and Peter Meerwald-Stadler. "Simplification of General Mixed Boolean-Arithmetic Expressions: GAMBA". In: *WORMA'23*. 2023.

[14] Rolf Rolles. "Unpacking Virtualization Obfuscators". In: *USENIX Conference on Offensive Technologies*. WOOT'09. 2009.

[15] Jonathan Salwan, Sébastien Bardin, and Marie-Laure Potet. "Symbolic deobfuscation: from virtualized code back to the original". In: *DIMVA*. 2018.

[16] Yongxin Zhou, Alec Main, and Gu et al. "Information Hiding in Software with Mixed Boolean-arithmetic Transforms". In: *Conference on Information Security Applications*. 2007.