Research-oriented internships, 4-6 months, CEA Paris-Saclay, France

Code-level Cybersecurity & Program Analysis: Vulnerabilities, Verification, Reverse

Keywords: software security, vulnerabilities, reverse & deobfuscation,
program analysis, formal methods, static analysis, symbolic execution, logic

The BINary-level SECurity research group (BINSEC) at CEA List has several open internship positions at the crossroad of software security, program analysis and formal methods, to begin *as soon as possible* at Paris-Saclay, France. Positions are 4-6 month long and can naturally *open the way to a doctoral work.* All these positions are articulated around the BINSEC open-source platform (`https://binsec.github.io`), which aims at providing automatic tools for low-level security analysis by adapting software verification methods initially developed for safety-critical systems.

Topic	Security, Logic and Verification, Program Analysis
Host	Commissariat à l'Énergie Atomique, Software Security Laboratory
Place	Paris-Saclay, France
Team	Binary-level security analysis
Advisor	Sébastien Bardin (`sebastien.bardin@cea.fr`)

Context. Several major classes of security analyses have to be performed on machine code, such as vulnerability analysis of mobile code or third-party components, deobfuscation or malware inspection. These analyses are very challenging, yet still relatively poorly tooled. Our long-term goal is to leverage recent advances in software verification, security analysis and artificial intelligence in order to propose efficient semantic tools for low-level security investigations.

Current topics. We propose several research directions, each one aiming at extending some recent work published in top tiers venue

- *Vulnerability detection at scale* [1, 3], with combination of cutting edges techniques such as symbolic execution, fuzzing and static analysis – the challenge here is to design *effective* combinations enjoying both precision and scalability for different classes of vulnerabilities;
- *Binary-level formal verification* of cryptographic implementations [2,4] with (variants of) symbolic execution – the challenge here is to handle advanced security properties such as non-interference or side channel leaks (timing, cache, power), as well as low-level micro-architectural behaviours such as speculation (Spectre) or faults (RawHammer);
- *Advanced reverse and certified decompilation*, through the combination of program analysis and artificial intelligence [5–7], with the ultimate goal of recovering legitimate high-level code equivalent to the original executable file.

More details on the topics will be happily provided! The list is not exhaustive, ask us if you have some project in mind.

All positions include theoretical research as well as prototyping (preferably in OCaml) and experimental evaluation. Results will be integrated in the open-source BINSEC platform.

Host. The BINary-level SECurity research group (BINSEC) of CEA List is a leading group in formal methods for low-level security, with regular publications in top-tier venues in security, formal methods and software engineering. We work in close collaboration with other French and international research teams, industrial partners and national agencies. CEA List is located in Campus Paris Saclay.

Requirements & application. We welcome curious and enthusiastic students with a solid background in Computer Science, both theoretical and practical. A good knowledge of functional programming (OCaml) is appreciated. Some experience in verification, security, logic or compilation would be great. Applicants should send an e-mail to Sébastien Bardin (`sebastien.bardin@cea.fr`) – including CV and motivation letter. **Deadline:** as soon as possible (first come, first served). Contact us for **more information**.

References

[1] Patrice Godefroid, Michael Y. Levin, and David A. Molnar. SAGE: whitebox fuzzing for security testing. *Commun. ACM*, 55(3):40–44, 2012.

[2] Lesly-Ann Daniel, Sébastien Bardin, and Tamara Rezk. Binsec/Rel: Efficient Relational Symbolic Execution for Constant-Time at Binary-Level In *2020 IEEE Symposium on Security and Privacy (S&P 2020)*.

[3] Sébastien Bardin, Manh-Dung Nguyen About Directed Fuzzing and Use-After-Free: How to Find Complex & Silent Bugs? Black Hat USA 2020

[4] Lesly-Ann Daniel, Sébastien Bardin, Tamara Rezk. Hunting the Haunter: Efficient Relational Symbolic Execution for spectre with Haunted RelSE. In the *28th Network and Distributed System Security Symposium (NDSS 2021)*

[5] Grégoire Menguy, Sébastien Bardin, Richard Bonichon, Cauim de Souza Lima. Search-based Local Blackbox Deobfuscation: Understand, Improve and Mitigate. In the *28th ACM Conference on Computer and Communications Security (CCS 2021)*

[6] Frédéric Recoules, Sébastien Bardin, Richard Bonichon, Laurent Mounier, Marie-Laure Potet. Get rid of inline assembly through verification-oriented lifting. In the *34th IEEE/ACM International Conference on Automated Software Engineering (ASE 2019)*.

[7] Sébastien Bardin, Robin David, and Jean-Yves Marion. Backward-bounded DSE: targeting infeasibility questions on obfuscated codes. In *2017 IEEE Symposium on Security and Privacy (S&P 2017)*.

[8] Benjamin Farinier, Sébastien Bardin, Richard Bonichon, Marie-Laure Potet. Model Generation for Quantified Formulas: A Taint-Based Approach. In *Proceedings of the 30th International Conference on Computer Aided Verification.* CAV 2018

[9] Guillaume Girol, Benjamin Farinier, Sébastien Bardin. Not All Bugs Are Created Equal, But Robust Reachability Can Tell The Difference. In the *33th International Conference on Computer Aided Verification.* CAV 2021