Master internship F/M: Type-based security properties assurance in the Rust-based Redox operating system

Le descriptif de l’offre ci-dessous est en Anglais

Type de contrat : Convention de stage

Niveau de diplôme exigé : Bac + 4 ou équivalent

Fonction : Stagiaire de la recherche

A propos du centre ou de la direction fonctionnelle

The Inria Centre at Rennes University is one of Inria’s eight centres and has more than thirty research teams. The Inria Centre is a major and recognized player in the field of digital sciences. It is at the heart of a rich PME and innovation ecosystem: highly innovative PMEs, large industrial groups, competitiveness clusters, research and higher education players, laboratories of excellence, technological research institute, etc.

Contexte et atouts du poste

The internship is proposed in the new cybersecurity research team Sushi focusing on the hardware/software interface. The advisors are operating system and cybersecurity experts from both academia and the french ministry of defense.

https://team.inria.fr/sushi/

Advisors:
• Louis Rilling (DGA, team Sushi) louis.rilling@irisa.fr
• Frédéric Tronel (CentraleSupélec, team Sushi) frederic.tronel@centralesuplec.fr

Mission confiée

Context:
Operating systems, especially their kernel, are critical software to build applications aiming at providing security properties. A vulnerability in the kernel opens a door for attackers to bypass the application logic, whatever the correctness of the application itself.

Getting assurance on security properties of operating systems services is known to incur high costs. This traditionally relies on careful design and heavy testing, with costs of up to 400% of a non-secure development effort [1]. While formal verification remains a challenge for this class of software, few projects achieved this goal for an up to 1000% cost [2].

We propose to explore a middle way to get assurance, relying on the compiler to check both of memory safety and higher-level, logical, security properties that should be encoded in the source code using types. It is expected that this reduces the debugging effort and it could make formal verification easier.

Keywords: Operating systems; Security; Programming languages; Rust; Redox.

Principales activités

An example of relevant techniques for operating system development is the typestate design pattern [3]. This technique gives assurance on the implementation of state machines, by assigning a dedicated type to each state of the state machine and encoding state machine transitions as functions converting from a source state to a target state. Invalid transitions are thus made impossible because the converting functions just do not exist. Session types [4] achieve a similar goal for protocols.

Although using types to achieve security has been known for decades, this practice using the programming language of an operating system is hardly explored. Current approaches rely on a separate language and its compiler to write annotations in the original source code and verify that the code satisfies the specified properties [5].

Yet applying typestate analysis on production source code like Linux shows that the security of such development could be improved if types were used to encode logical properties [6]. On the other hand, using the type system of the programming language to ensure functional properties is the topic of
ongoing research [7].

The Rust programming language is especially a good candidate as it is strongly- and statically-typed, by
design the compiler checks the memory safety of programs, and it is the basis of several open-source
operating system projects, including Redox 1, which explicitly targets security. Moreover successful Rust-
implementations were demonstrated for typestates [8] and session types [9].

One of the mechanisms provided by Redox to achieve security is called schemes. Schemes generalize
Unix “everything is a file” notion to “everything is a URL”. Using namespace reduction of accessible
schemes (the Redox-specific setrens system call), open handles can then be used as capabilities [10].

After a study of the state of the art, the suggested work plan is as follows:
- First it should be shown how the typestate design pattern helps getting assurance on
isolating clients of a scheme providing access to a cryptographic resource. To this end a
representative userspace scheme daemon will be implemented in Rust for Redox.
- Second a comparative study should be done on generically implementing resource read-
and write- access restriction in the kernel, using traditional attribute-based checks on one
side and type-based segregation of handles on the other side. During the study the two
approaches will be implemented and experimentally compared.

References
[1] Elaine Venson, Xiaomeng Guo, Zidi Yan, and Barry Boehm. Costing secure software
development: A systematic mapping study. In Proceedings of the 14th International Con-
Association for Computing Machinery.
Kolanski, and Gernot Heiser. Comprehensive formal verification of an os microkernel.
171, 1986.
typing system. In Costas Halatsis, Dimitrios Maritsas, George Philokyprou, and Sergios
Theodoridis, editors, PARLE'94 Parallel Architectures and Languages Europe, pages 398–
Type-driven verification of non-functional properties. In Proceedings of the 21st Interna-
tional Symposium on Principles and Practice of Declarative Programming, PPDP ’19, New
York, NY, USA, 2019. Association for Computing Machinery.
https://www.redox-os.org/
types for rust. In Proceedings of the 11th ACM SIGPLAN Workshop on Generic Program-
ing, WGP 2015, page 13–22, New York, NY, USA, 2015. Association for Computing
Machinery.
[10] Jack B. Dennis and Earl C. Van Horn. Programming semantics for multiprogrammed com-

Compétences
Required skills: System programming; curiosity.
Appreciated but not mandated skills: Deep understanding of OSes; Programming in Rust.

Rémunération
Bonus according to current rate

Informations générales
- Theme/Domaine : Sécurité et confidentialité
- Ville : Rennes
- Centre Inria : Centre Inria de l'Université de Rennes
- Date de prise de fonction souhaitée : 2024-02-01
- Durée de contrat : 5 mois
- Date limite pour postuler : 2024-01-15

Contacts
- Equipe Inria : SUSHI (DGD-S)
A propos d'Inria

Inria est l'institut national de recherche dédié aux sciences et technologies du numérique. Il emploie 2600 personnes. Ses 215 équipes-projets agiles, en général communes avec des partenaires académiques, impliquent plus de 3900 scientifiques pour relever les défis du numérique, souvent à l'interface d'autres disciplines. L'institut fait appel à de nombreux talents dans plus d'une quarantaine de métiers différents. 900 personnels d'appui à la recherche et à l'innovation contribuent à faire émerger et grandir des projets scientifiques ou entrepreneuriaux qui impactent le monde. Inria travaille avec de nombreuses entreprises et a accompagné la création de plus de 200 start-up. L'institut s'efforce ainsi de répondre aux enjeux de la transformation numérique de la science, de la société et de l'économie.

**Attention:** Les candidatures doivent être déposées en ligne sur le site Inria. Le traitement des candidatures adressées par d'autres canaux n'est pas garanti.

Consignes pour postuler

**Sécurité défense :**
Ce poste est susceptible d'être affecté dans une zone à régime restrictif (ZRR), telle que définie dans le décret n°2011-1425 relatif à la protection du potentiel scientifique et technique de la nation (PPST). L'autorisation d'accès à une zone est délivrée par le chef d'établissement, après avis ministériel favorable, tel que défini dans l'arrêté du 03 juillet 2012, relatif à la PPST. Un avis ministériel défavorable pour un poste affecté dans une ZRR aurait pour conséquence l'annulation du recrutement.

**Politique de recrutement :**
Dans le cadre de sa politique diversité, tous les postes Inria sont accessibles aux personnes en situation de handicap.