Combiner la vérification déductive avec l'analyse de forme

Téo Bernier, Yani Ziani, Nikolai Kosmatov, Frédéric Loulergue

Combiner la vérification déductive avec l'analyse de forme *

Téo Bernier[1], Yani Ziani[1,2], Nikolai Kosmatov[1], and Frédéric Lourlergue[2]

[1]Thales Research & Technology, Palaiseau, France,
firstname.lastname@thalesgroup.com
[2]Univ. Orléans, INSA Centre Val de Loire, LIFO EA 4022
frederic.loulergue@univ-orleans.fr

Contexte. Des outils de vérification déductive ont pu être utilisés avec succès dans de nombreuses études de cas [2] pour prouver un large panel de propriétés de sûreté, sécurité, ainsi que des propriétés fonctionnelles. De tels outils rencontrent souvent des difficultés à procéder à des preuves automatiques dans des codes manipulant des *structures de données récursives* (par exemple les listes chaînées, les arbres, etc.), en particulier, du fait des modèles mémoire complexes dont ils ont besoin. L'utilisateur doit alors guider la preuve par des lemmes prouvés interactivement, des assertions, etc. Les outils d'interprétation abstraite basés sur la logique de séparation et l'analyse de forme [3] peuvent raisonner efficacement sur de telles structures, mais ne peuvent typiquement pas gérer de larges classe de propriétés. L'article [1] résumé ici présente de nouvelles idées et les premiers résultats dans une tentative de combinaison de ces deux techniques pour profiter des capacités de chacune.

Approche. Nous présentons une démarche de vérification qui combine un outil de vérification déductive populaire pour les programmes C, FRAMA-C/WP [4], avec un outil d'analyse de forme, MEMCAD [5]. L'idée principale est de prouver des propriétés structurelles et des propriétés de séparation de la mémoire dans MEMCAD, puis de les admettre dans FRAMA-C/WP de sorte à accroître le niveau d'automatisation de ce dernier et de surmonter certaines de ses limitations.

Nous appliquons cette approche à une étude de cas concrète utilisant les listes chaînées : quelques fonctions (légèrement simplifiées) de tpm2-tss [1], une librairie populaire de communication avec le Trusted Platform Module (TPM). Des travaux récents [6] ont montré que la vérification déductive des fonctions manipulant les

*Cette soumission est un résumé étendu d'un article [1], accepté à FASE 2024.

1. https ://github.com/tpm2-software/tpm2-tss

listes chaînées de cette librairie était relativement compliquée, et a demandé l'ajout de nombreux lemmes et assertions.

Contributions. Nous proposons une technique de vérification mixte qui utilise la vérification déductive et l'analyse de formes, que nous illustrons avec FRAMA-C/WP et MEMCAD sur une fonction manipulant des listes chaînées, ainsi qu'une étude de cas concluante sur un ensemble de fonctions de la librairie tpm2-tss, une implémentation open-source très utilisée de la TPM Software Stack (TSS) [2] conçue pour accéder au Trusted Platform Module (TPM). Cette librairie utilise des listes chaînées pour stocker et utiliser les ressources TPM, telles que les objets échangés avec le TPM. Les cellules de liste y sont allouées dynamiquement, et nous avons appliqué des simplifications aux structures de données utilisées pour les cellules de liste (et leur traitement).

Résultats. Pour les fonctions considérées, la preuve utilisant seulement FRAMA-C/WP nécessitait l'usage et la définition de 14 lemmes et a pris 4m50s. Grâce à la combinaison de WP/MEMCAD, nous avons pu nous passer de 5 de ces lemmes et de $\sim$45 annotations en ACSL. Nous avons donc eu besoin de seulement 9 lemmes et la preuve complète n'a pris qu'1min47s (la partie MEMCAD n'ayant nécessité que moins d'une seconde).

Conclusion et travaux futurs. Nous avons montré la pertinence de notre approche en l'appliquant avec succès à une étude de cas de code réel, dans lequel les propriétés de séparation et les invariants structurels ont pu être plus facilement prouvés par l'analyse de formes, puis utilisés comme hypothèses pour la vérification déductive, qui a pu être ainsi dirigée vers d'autres propriétés.

Ces travaux ouvrent des perspectives de recherche intéressantes : l'automatisation de la technique de vérification proposée en incluant la génération coordonnée des hypothèses et des assertions, la preuve de correction, la conception d'un mécanisme de spécification (de plus haut-niveau d'abstraction) commun pour les structures de données récursives avec une traduction automatique en propriétés appropriées pour MEMCAD et FRAMA-C, ainsi que l'évaluation sur d'autres études de cas pertinentes.

Remerciements. Ce travail a été partiellement soutenu par l'ANR (bourses ANR-22-CE39-0014, ANR-22-CE25-0018). Les travaux du deuxième auteur ont été partiellement financés par une bourse de thèse du Ministère de la Défense. Nous remerçions Allan Blanchard, Laurent Corbin, Loïc Correnson, Daniel Gracia Pérez et Xavier Rival pour les discussions enrichissantes.

2. `https://trustedcomputinggroup.org/work-groups/software-stack/`

Références

[1] T. BERNIER, Y. ZIANI, N. KOSMATOV et F. LOULERGUE. « Combining Deductive Verification with Shape Analysis ». In : *Proc. of the 27th International Conference on Fundamental Approaches to Software Engineering (FASE 2024), Held as Part of the European Joint Conferences on Theory and Practice of Software (ETAPS 2024)*. LNCS. To appear. Springer, avr. 2024.

[2] R. HÄHNLE et M. HUISMAN. « Deductive Software Verification : From Pen-and-Paper Proofs to Industrial Tools ». In : *Computing and Software Science - State of the Art and Perspectives*. T. 10000. LNCS. Springer, 2019, p. 345-373.

[3] D. DISTEFANO, P. W. O'HEARN et H. YANG. « A Local Shape Analysis Based on Separation Logic ». In : *12th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06)*. T. 3920. LNCS. Springer, 2006, p. 287-302.

[4] F. KIRCHNER, N. KOSMATOV, V. PREVOSTO, J. SIGNOLES et B. YAKOBOWSKI. « Frama-C : A software analysis perspective ». In : *Formal Asp. Comput.* 27.3 (2015), p. 573-609.

[5] P. SOTIN et X. RIVAL. « Hierarchical Shape Abstraction of Dynamic Structures in Static Blocks ». In : *10th Asian Symposium on Programming Languages and Systems (APLAS'12)*. T. 7705. LNCS. Springer, 2012, p. 131-147.

[6] Y. ZIANI, N. KOSMATOV, F. LOULERGUE, D. GRACIA PÉREZ et T. BERNIER. « Towards Formal Verification of a TPM Software Stack ». In : *Proc. of the 18th International Conference on integrated Formal Methods (iFM 2023)*. T. 14300. LNCS. Springer, nov. 2023, p. 93-112.