Language-Support for Correct and Reliable Enforcement of Access Control Policies

Abstract

Peter Amthor*
Technische Universität Ilmenau

Keywords: *operating systems · security policy · access control · security engineering · domain-specific language · Rust*

As highlighted by security-focused OS mechanisms such as SELinux [10], Solaris Trusted Extensions [5], or TaintDroid [4], a mandatory access control (AC) policy is the fundamental corner stone of any security guarantees. A correct and reliably enforced AC policy provides a solid and reliable defense against privilege escalation attacks, i. e. threats such as root kit malware and ransomware. Both requirements are covered in the well-known reference monitor (RM) principle [2]: while the *verifiability* requirement demands both correctness by design and formally provable properties, the *total mediation* and *tamperproofness* requirements focus on reliable enforcement of a policy.

Such abstract requirements contrast with a practical process of design, specification and implementation of an AC policy, which inevitably introduces the human element of de-abstraction and decision-making. To minimize the impact of this, specialized languages used for these tasks have to satisfy a diverse range of requirements:

Adequate abstractions: The semantics and granularity of abstractions used to specify a policy, e. g. user attributes, object classes, domains etc., should match those used in the enforcing system. A widely adopted approach is the use of a specification language for attribute-based access control (ABAC).

Verifiability: The policy logic should be verifiable against formal properties, e. g. privilege escalation *safety* [6] or *workflow satisfiability* [3]. This inevitably involves representing the policy in an adequately expressive formal calculus.

Ergonomics: Syntax, semantics and idiomatics of a language used to implement the policy, e. g. in an OS kernel module or a server process, should constructively avoid errors that invalidate any previously achieved correctness guarantees.

Reliable enforcement: As an interface to enforce a policy, a generalized runtime environment (RTE) is required which enforces both *total mediation* and *tamperproofness* of the policy implementation, independent of their OS- and application-specific specification.

As becomes apparent, all four requirements depend on languages to represent an AC policy that significantly differ in level of abstraction, expressiveness, syntax, and semantics. This results in several, possibly error-prone translation steps. Paradoxically, such translations counter the very goal of the individual languages in this process, since they again allow errors to be introduced through manual interpretation and rewriting.

In our work to be presented based on this abstract, we argue for an approach to AC policy engineering and implementation that aims at two goals: (1) whenever possible, translations between heterogeneous policy representations should be done automatically; (2) whenever inevitable, manual translations effort should be as low as possible.

Based on original contributions towards these goals, we illustrate how we can already model, specify and translate an ABAC policy to an actual implementation in the Rust programming language [7]. We introduce DynaMo, a novel policy specification language, which features a syntax and semantics derived from mathematical notation conventions of a flexible formal ABAC calculus [8]. Based on a state-machine simulation implemented for this calculus [1], it enables automated analyses of dynamic security properties. We further present `dmo2rs` as ongoing work: a transpiler from DynaMo to the Rust programming language for correct policy implementation. We conclude with `dabac-rs` [9], a prototype of a policy-neutral RTE, to be integrated in systems software as a ready-made Rust crate.

* peter.amthor@tu-ilmenau.de

References

[1] Peter Amthor and Marius Schlegel. Towards Language Support for Model-based Security Policy Engineering. In Pierangela Samarati, Sabrina De Capitani di Vimercati, Mohammad S. Obaidat, and Jalel Ben-Othman, editors, *Proceedings of the 17th International Conference on Security and Cryptography*, SECRYPT 2020, pages 513–521. INSTICC, SciTePress, 2020.

[2] James P. Anderson. Computer Security Technology Planning Study. Technical Report ESD-TR-73-51, Air Force Electronic Systems Division, Hanscom AFB, Bedford, MA, USA, 1972. Also available as Vol. I, DITCAD-758206. Vol. II DITCAD-772806.

[3] Pierre Berge, Jason Crampton, Gregory Gutin, and Remi Watrigant. The Authorization Policy Existence Problem. In *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*, CODASPY '17, pages 163–165, New York, NY, USA, 2017. ACM.

[4] William Enck, Peter Gilbert, Byung-Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI '10, pages 1–6, Berkeley, CA, USA, 2010. USENIX Association.

[5] Glenn Faden. Solaris Trusted Extensions – Architectural Overview, April 2006. Sun/Oracle White Paper.

[6] S. Jha, S. Sural, V. Atluri, and J. Vaidya. An Administrative Model for Collaborative Management of ABAC Systems and Its Security Analysis. In *IEEE 2nd International Conference on Collaboration and Internet Computing*, CIC 2016, pages 64–73. IEEE Press, 2016.

[7] Nicholas D. Matsakis and Felix S. Klock. The rust language. In *Proc. 2014 ACM SIGAda Annual Conference on High Integrity Language Technology*, HILT '14, page 103–104, New York, NY, USA, 2014. Association for Computing Machinery.

[8] Marius Schlegel and Peter Amthor. The Missing Piece of the ABAC Puzzle: A Modeling Scheme for Dynamic Analysis. In Sabrina De Capitani di Vimercati and Pierangela Samarati, editors, *Proceedings of the 18th International Conference on Security and Cryptography*, SECRYPT 2021, pages 234–246. INSTICC, SciTePress, 2021.

[9] Marius Schlegel and Peter Amthor. Putting the Pieces Together: Model-based Engineering Workflows for Attribute-based Access Control Policies. In Sabrina De Capitani di Vimercati and Pierangela Samarati, editors, *18th International Conference on Security and Cryptography, SECRYPT 2021, Revised Selected Papers*, volume 1795 of *CCIS*. Springer International Publishing, 2022.

[10] Stephen D. Smalley, Chris Vance, and Wayne Salamon. Implementing SELinux as a Linux Security Module. Technical Report 01-043, NAI Labs, May 2002.