Alacrity: A DSL for Simple, Formally-Verified DApps

Jay McCarthy

Head of Research, Alacris

Associate Professor, University of Massachusetts Lowell

Introduction

Alacrity is a domain-specific language for writing decentralized applications. It provides automatic solutions to a few key problems faced by blockchain developers: verifying that the DApp is trustworthy, ensuring the smart contract is consistent with client-side software, and abstracting over different blockchains.

Alacrity programs embed descriptions of properties about the behavior of the application: safety properties assert mistakes do not occur, and liveness properties assert desired outcomes do occur. The compiler automatically adds soundness properties that assert the application does not violate fundamental expectations of all DApps, such as that they conserve resources. The compiler automatically verifies correctness of these properties using the Z3 SMT theorem prover without intervention from programmers. This ensures that Alacrity programs are not susceptible to attacks that steal their resources, and ensures that untrusting participants can rely on the integrity and validity of the Alacrity program.

Alacrity programs incorporate the client-side behavior of participants, as well as on-chain behavior of the contract. The Alacrity compiler uses End-Point Projection to extract programs for each party and the contract, while guaranteeing each side makes the same assumptions about application state and communication protocols. This ensures that attacks do not exist that exploit the slightly different semantics of blockchain virtual machines and client-side programming languages.

Alacrity uses a blockchain-agnostic model of computation that allows programs to target different chains, including scaling solutions. This ensures that DApps can be designed independently of the deployment details and not be tied to the particular vagaries of any one platform.

The core philosophy of Alacrity is to design a highly constrained programming language that makes it easy to automatically prove the structural components of desirable properties about DApps, and makes it possible to easily prove the user-specific components of those properties. This is in contrast to designing an unconstrained language and providing a novel proving technique specialized for decentralized applications.

In this article, we take walk-through of an example Alacrity program and show how Alacrity performs each function.

Main Content

Please read this article in full at

github.com

AlacrisIO/alacrity/blob/f1c0326ec5ebfe8c2b0fd539378071d34583947d/docs/paper.md

Alacrity: A DSL for Simple, Formally-Verified DApps

Jay McCarthy jay@alacris.io

Head of Research, Alacris

Associate Professor, University of Massachusetts Lowell

Introduction

Alacrity is a domain-specific language for writing decentralized applications. It provides automatic solutions to a few key problems faced by blockchain developers: verifying that the DApp is trustworthy, ensuring the smart contract is consistent with client-side software, and abstracting over different blockchains.

Alacrity programs embed descriptions of properties about the behavior of the application: safety properties assert mistakes do not occur, and liveness properties assert desired outcomes do occur. The compiler automatically adds soundness properties that assert the application

Please leave comments here or on Github, whichever is convenient for you.

[…]

Conclusion

[…]

Although Alacrity is usable today, it is a work-in-progress with room for growth and development. You can start using it today by visiting AlacrisIO/alacrity. We will be running a workshop in Boston in Fall 2019. Please visitalacris.io or follow us on Twitter @alacrisos

or Telegram at alacrisio for more information!