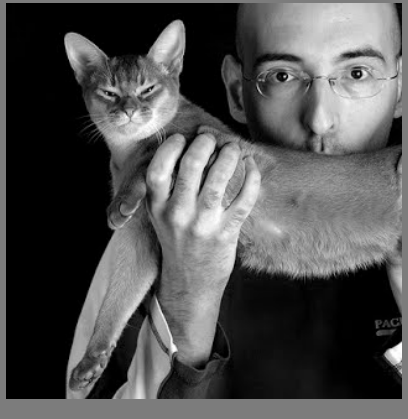
TALKS @ LASIGE driven by excellence

Fri
4
December
2020

BITCOIN CONTRACTS: SECURE COMPILATION, ANALYSIS, AND EXPRESSIVENESS



Speaker
Massimo Bartoletti
Affiliation
University of Cagliari

When
December 4th 15h00

Where
Zoom

Abstract

Although Bitcoin is mainly used to exchange crypto-currency, its blockchain and consensus mechanism can also be exploited to execute smart contracts, allowing mutually untrusted parties to exchange crypto-assets according to pre-agreed rules. To this purpose, Bitcoin features a non Turing-complete script language, which is used to specify the redeem conditions of transactions. This is a simple language of expressions, without loops or recursion. To write complex smart contracts, one needs to suitably combine transactions: in this approach, executing a contract amounts to appending sequences of transactions in a given order.

A drawback of this approach is that the complexity of writing smart contracts grows quickly in the number of transactions needed to implement it. Reasoning about the correctness of these contracts is even harder: one would have to consider computational adversaries who interact with the blockchain, only being constrained to use PPTIME algorithms. To overcome these issues we have proposed BitML, a high-level DSL for smart contracts with a computationally sound compiler to Bitcoin transactions.

The computational soundness property allows us to reason about contracts at the symbolic level of the BitML semantics. We exploit this possibility to investigate a landmark property of contracts, called liquidity, which ensures that funds never remain frozen within a contract. Liquidity is a relevant issue, as witnessed by a recent attack to the Ethereum Parity Wallet, which has frozen ~160M USD within the contract, making this sum unredeemable by any user.

We develop a static analysis for liquidity of BitML contracts. This is achieved by first devising a finite-state, safe abstraction of infinite-state semantics of BitML, and then model-checking this abstraction.

We conclude by discussing a few open issues: in particular, how to enhance the expressiveness of Bitcoin contracts via minor extensions of the Bitcoin script language, and how to reduce the cost of executing contracts.