Converging two directions of program verification: deductive verification meets smart types for smart contracts

António Ravara

Informatics Department
Faculty of Sciences and Technology
NOVA Lisbon

11 February 2022

NOVALINCS
LABORATORY FOR COMPUTER SCIENCE AND INFORMATICS

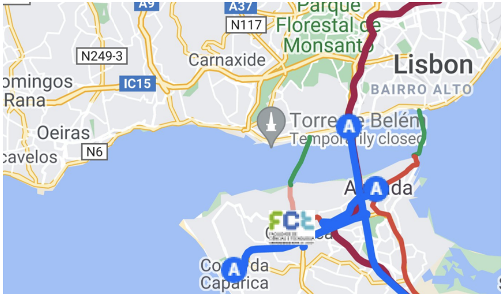
Where are we?

Lab for Computer Science and Informatics

https://nova-lincs.di.fct.unl.pt/the-center

In NOVA School of Science and Technology

The Faculty of Sciences on NOVA University of Lisbon
The Research Lab of the Department of Informatics



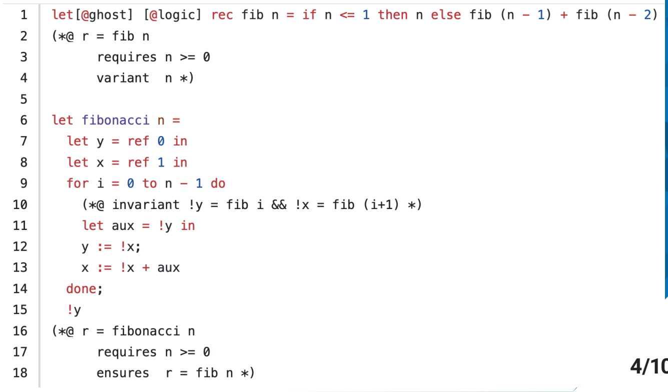
What are we doing?

Cameleer: A Deductive Verification Tool for OCaml Programs

https://mariojppereira.github.io/cameleer.html

Lead by Mário Pereira

1. Add assertions to OCaml
2. Get most discharged by Why3
3. Interactively prove the challenging ones



```
let[@ghost] [@logic] rec fib n = if n <= 1 then n else fib (n - 1) + fib (n - 2)
(*@ r = fib n
      requires n >= 0
      variant  n *)

let fibonacci n =
  let y = ref 0 in
  let x = ref 1 in
  for i = 0 to n - 1 do
    (*@ invariant !y = fib i && !x = fib (i+1) *)
    let aux = !y in
    y := !x;
    x := !x + aux
  done;
  !y
(*@ r = fibonacci n
      requires n >= 0
      ensures  r = fib n *)
```

Cameleer: limitation and research questions

WhyML type system does not allow mutable recursive data structures

To support reasoning about such OCaml programs,

- how to combine with other proof assistants, like CFML and Vyper?
- What kind of memory model could/should one add to the Why3 logic?

Java Typestate Checker

`https://github.com/jdmota/java-typestate-checker`

Statically check that

1. class methods are called in a prescribed order, specified in a protocol
2. object protocols are completed
3. absence of null pointer errors

```
typestate LineReaderProtocol {
  Init = {
    Status open(String): <OK: Open, ERROR: end>
  }
  Open = {
    boolean eof(): <true: Close, false: Read>,
    void close(): end
  }
  Read = {
    String read(): Open,
    void close(): end
  }
  Close = {
     void close(): end
  }
}
```

Java Typestate Checker: a limitation and research questions

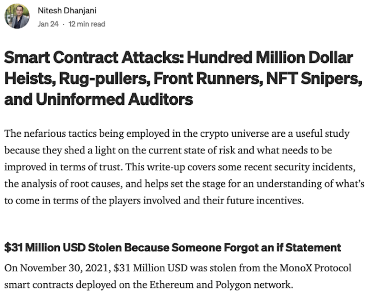
Session types control shared resources with a linear discipline

To support reasoning about structures like collections,

- how to describe and control collectively each individual resource?
- how to safely relax linearity, mantaining static assurances of protocol compliance and completion?

Challenges requiring to put it all together

Exploitable vulnerabilities in smart contracts



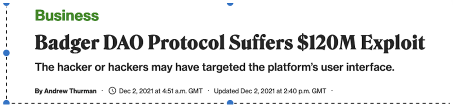
Nitesh Dhanjani
Jan 24 · 12 min read

Smart Contract Attacks: Hundred Million Dollar Heists, Rug-pullers, Front Runners, NFT Snipers, and Uninformed Auditors

The nefarious tactics being employed in the crypto universe are a useful study because they shed a light on the current state of risk and what needs to be improved in terms of trust. This write-up covers some recent security incidents, the analysis of root causes, and helps set the stage for an understanding of what's to come in terms of the players involved and their future incentives.

$31 Million USD Stolen Because Someone Forgot an if Statement

On November 30, 2021, $31 Million USD was stolen from the MonoX Protocol smart contracts deployed on the Ethereum and Polygon network.

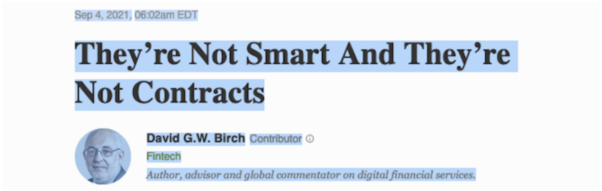


Business

Badger DAO Protocol Suffers $120M Exploit

The hacker or hackers may have targeted the platform's user interface.

By Andrew Thurman · Dec 2, 2021 at 4:51 a.m. GMT · Updated Dec 2, 2021 at 2:40 p.m. GMT ·



Sep 4, 2021, 06:02am EDT

They're Not Smart And They're Not Contracts

David G.W. Birch Contributor
Fintech
Author, advisor and global commentator on digital financial services.

Smarter smart contracts... controlled by (software) contracts

Better programming languages

- Lightweight (graphical) annotations, to declare protocols
- Code generation from scribbled specifications
- Verification at compile-time
    - behavioural types check protocol conformance
    - deductive verification / liquid types check assertions

Monitored run-time execution

- Attackers don’t "respect the rules"
- Specs can be used to control the execution of contracts
- Enforcement at run-time