Sereum: Protecting Existing Smart Contracts Against Re-Entrancy Attacks,

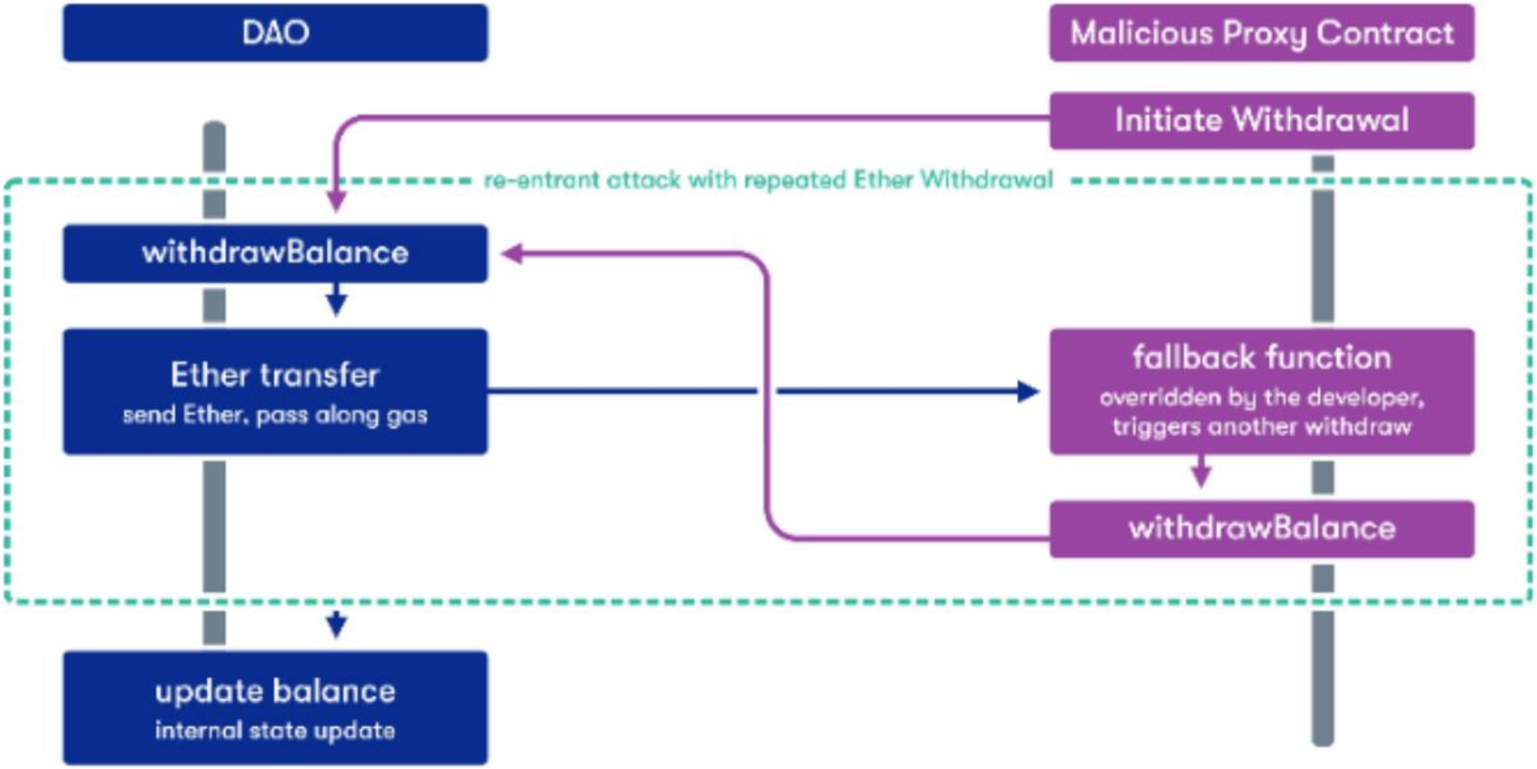
M Rodler, W Li, GO Karame, L Davi, NDSS 2019

CS 595 Paper Presentation
Yanfeng Qu

ILLINOIS INSTITUTE
OF TECHNOLOGY

Re-Entrancy Attacks - DAO Attacks

- In Ethereum, when there is a function call
  - The caller has to wait for the call to finish
  - A malicious callee might take advantage of this



Picture Source: https://cointelegraph.com.br/news/the-vulnerabilities-of-smart-contracts

Challenges

- The code of a smart contract is expected to be immutable after deployment
- Smart contract owners are anonymous, i.e., responsible disclosure is usually infeasible
- Existing approaches are mostly performing offline analysis and are susceptible to missing unknown runtime attack patterns.

Related works - Reentrancy Attack Detection

Static analysis

- SmartCheck [Tikhomirov et al., CCS18]
- Securify [Tsankov et al., CCS18]

Runtime Checking

- ECFChecker [Grossman et al., POPL18]

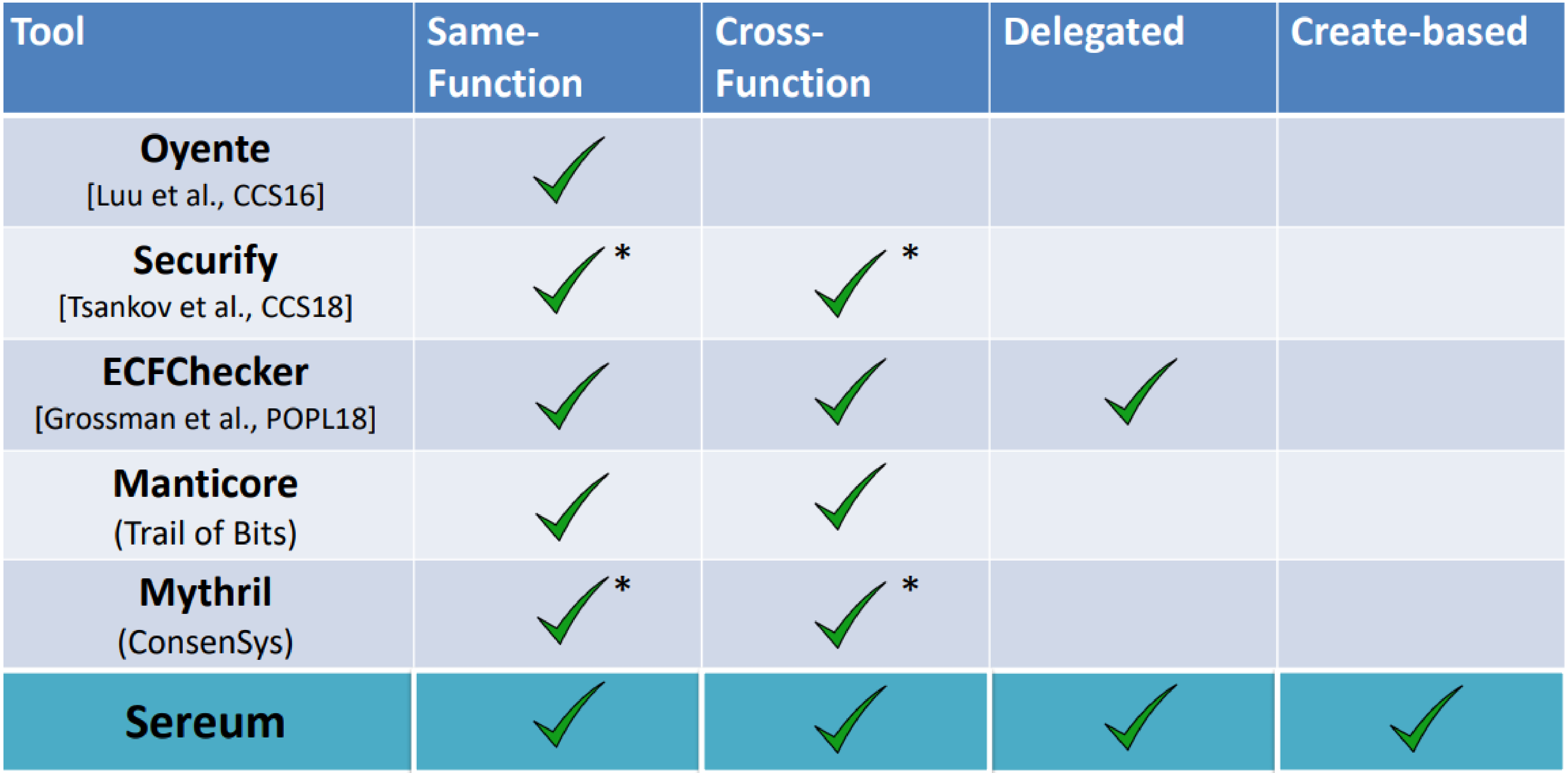
Symbolic execution

- Oyente [TLuu et al., CCS16]
- Summary-based Symbolic [Yu et al., ASE20]

Verification

- ZEUS [Kalra et al., NDSS18]
- RA [TLuu et al., CCS16]

Overview on Re-Entrancy Detection



Tool	Same-Function	Cross-Function	Delegated	Create-based
Oyente [Luu et al., CCS16]	✓			
Securify [Tsankov et al., CCS18]	✓ *	✓ *		
ECFChecker [Grossman et al., POPL18]	✓	✓	✓	
Manticore (Trail of Bits)	✓	✓		
Mythril (ConsenSys)	✓ *	✓ *		
Sereum	✓	✓	✓	✓

* Conservative policy with high number of FP

Picture Source: https://www.ndss-symposium.org/wp-content/uploads/ndss2019_09-3_Rodler_slides.pdf

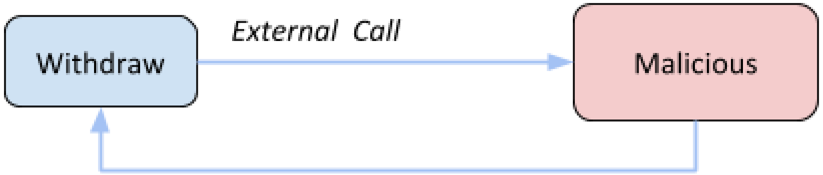
Sereum

- Sereum protects existing, deployed contracts against re-entrancy attacks in a backwards compatible way based on run-time monitoring and validation

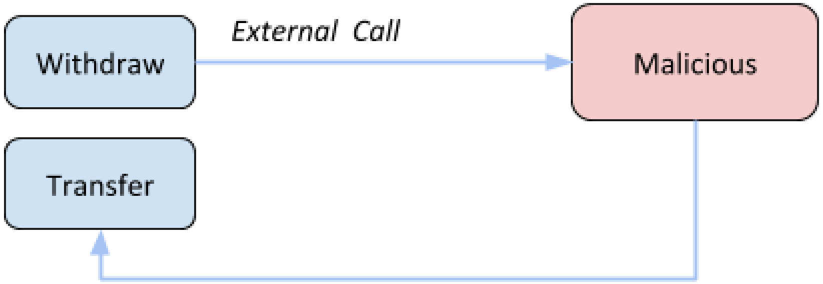
- Contributions
  - 3 more types of reentrancy attacks
  - Design and implementation of Sereum (Secure Ethereum)
  - Performance evaluation: performance overhead, FP

4 Types of Reentrancy Attacks

- Single Function Reentrancy
  - The DAO Attack
  - the fallback function recursively calling withdraw

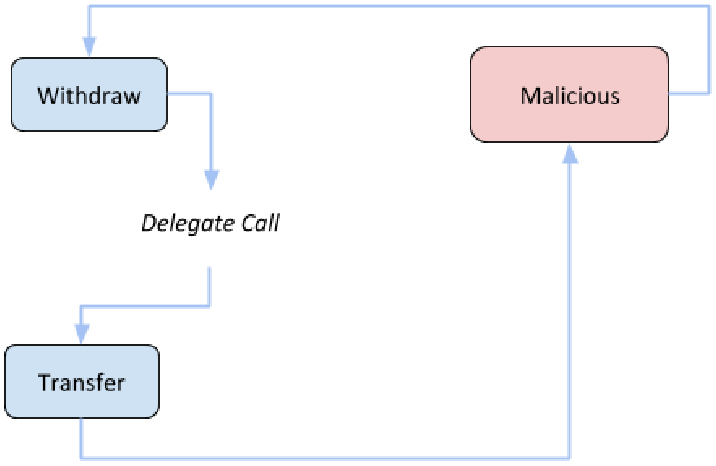


- Cross-function Reentrancy
  - the fallback function call function transfer

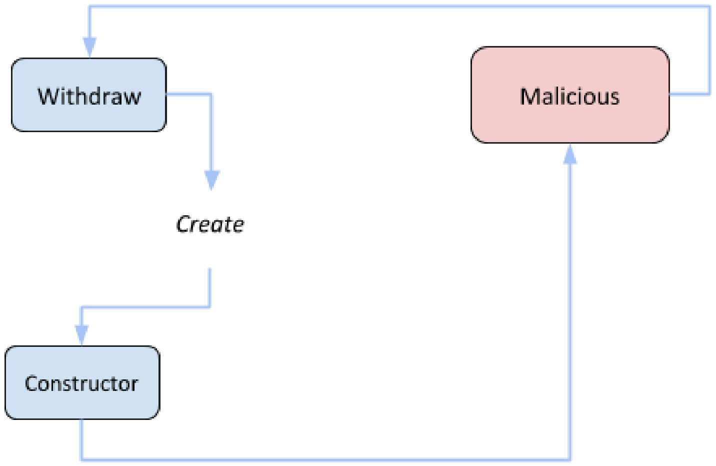


4 Types of Reentrancy Attacks

- Delegated Reentrancy
  - a contract invokes another contract as a library

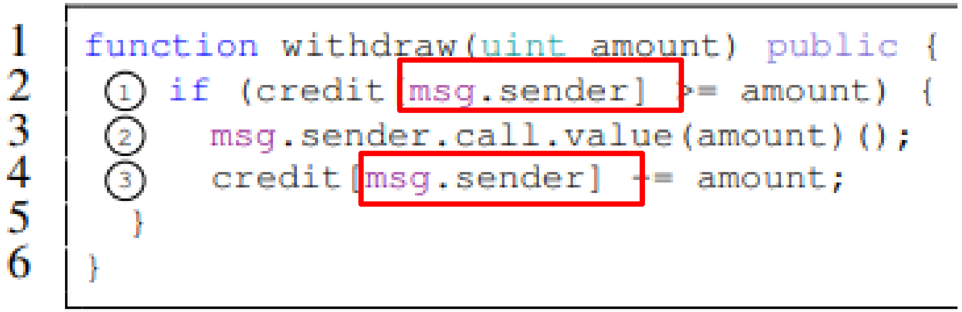


- Create-Based Reentrancy
  - issue further calls in its constructor to other contracts, including malicious contracts



Sereum Approach and Architecture

- Sereum Approach



```
function withdraw(uint amount) public {
 ① if (credit[msg.sender] >= amount) {
 ②   msg.sender.call.value(amount)();
 ③   credit[msg.sender] -= amount;
   }
}
```

Mark variables that influence branching decisions as critical

Prevent further updates with write locks

- Sereum Architecture
  - Attack Detector

    Enforcement: Transaction roll-back on detected attack

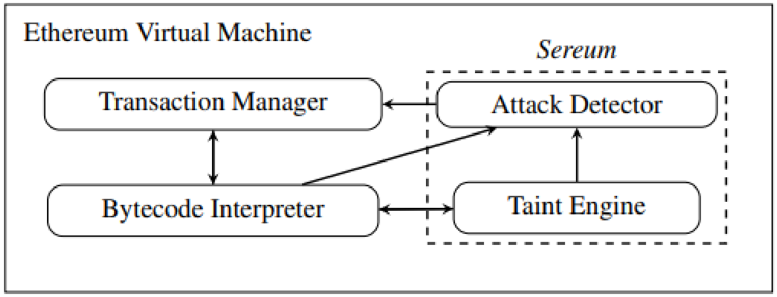
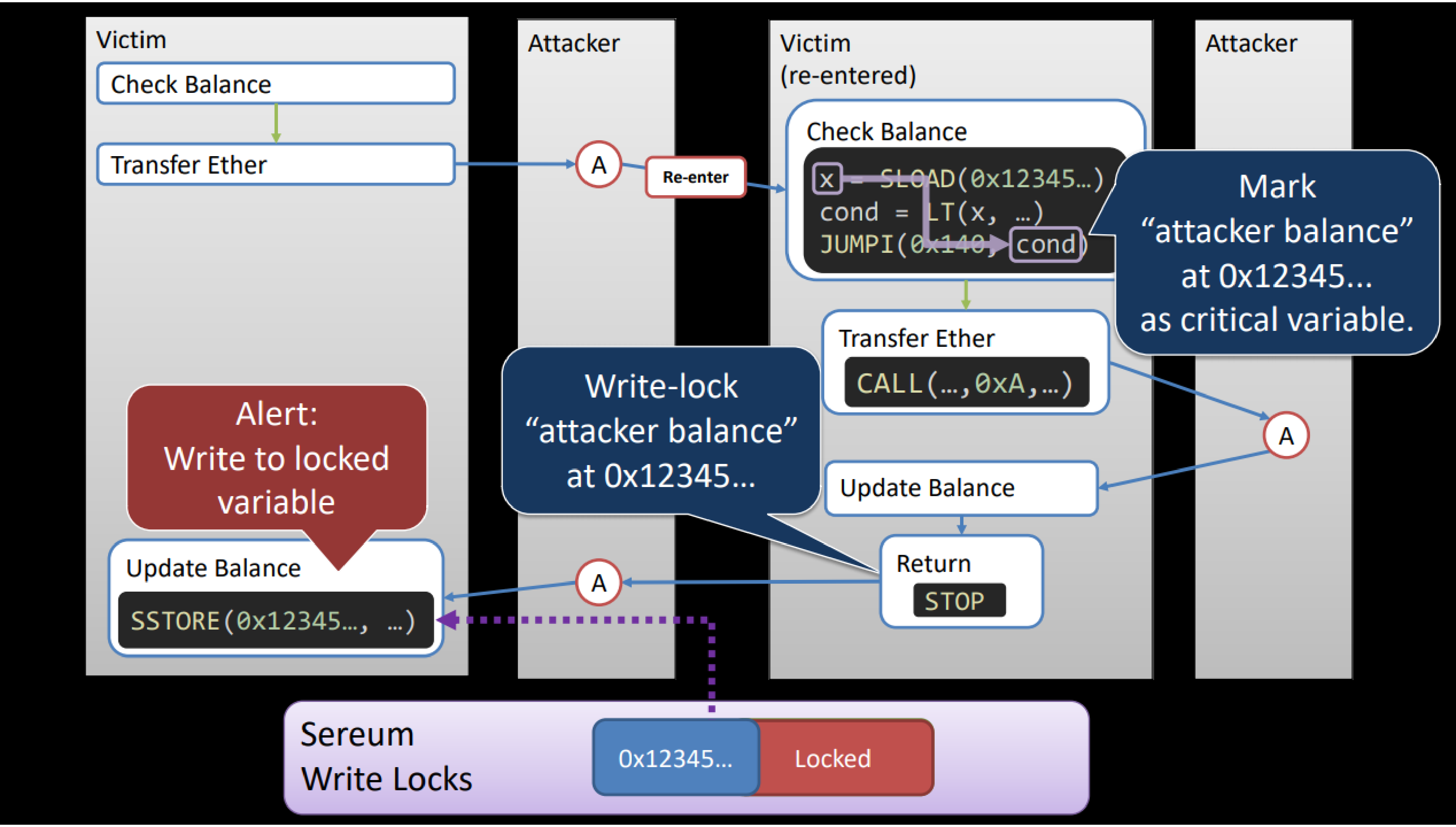


Figure 5. Architecture of enhanced EVM with run-time monitoring.

Picture Source: [M Rodler,19]

Write Locks



Picture Source: https://www.ndss-symposium.org/wp-content/uploads/ndss2019_09-3_Rodler_slides.pdf

Implementation and Evaluation

- Develop a working prototype system in EVM (Go ethereum)
  - the taint engine and the reentrancy attack detector

- Evaluation
  - Memory overhead 9767 MB
  - Performance overhead 217.6 ms (9.6%)

- Evaluation on 4.5 Million Ethereum blocks; Successful detection of the DAO incident
  - 50k flagged transactions
  - FP rate: 0.06%

Limtations

- False Positive Causes
  - Manual Re-Entrancy Locking
  - …
- Runtime checking

Thank you

- Questions