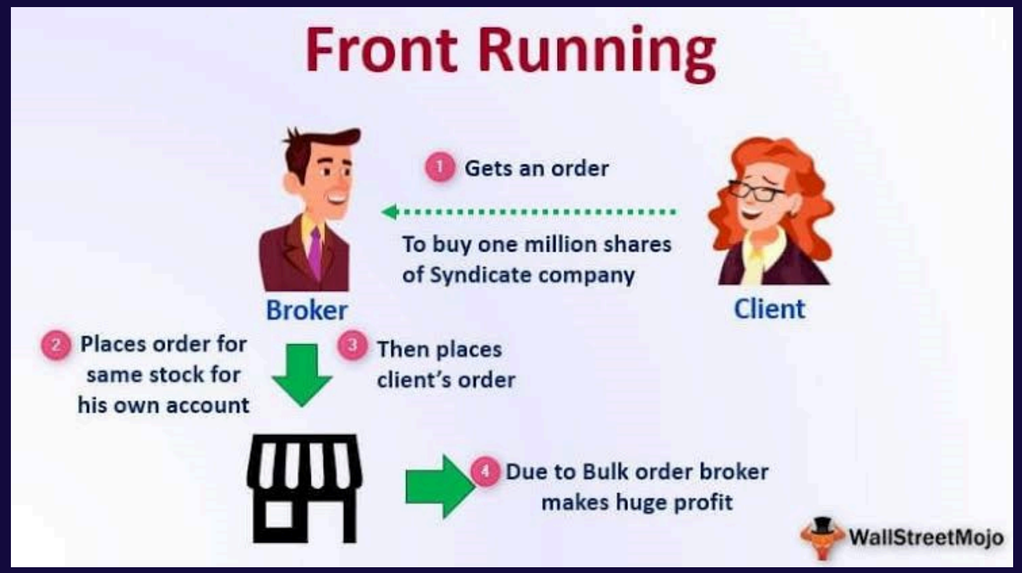
# Blockchain Security Vulnerabilities- Analysis of smart contract exploits (Reentrancy, Front-running).

Mark Lopes(9913)
Vivian Ludrick(9914)
Rohit Patra(9918)

Paper 1

# Combatting Front-Running in Smart Contracts

## What is Front-Running?




**Front-running is when an attacker sees a pending transaction and submits their own similar or conflicting transaction with a higher gas fee to get processed first.**

**User A (Victim):**
**wants to sell 100 TokenX for TokenY**
**current exchange rate is 1 TokenX = 1 TokenY**
**A submits the transaction**

**User B (Attacker):**
**Monitors the blockchain mempool**
**Sees A's large swap which will increase the price of TokenY**
**B quickly submits their own transaction to buy TokenY before A, with a higher gas fee**

Paper 1

# Algorithms used to combat front running

## Attack Mining Algorithm

They created a new model for what counts as a front-running attack:
If an A profits AND the B loses, it’s an attack.

They used smart pruning techniques to skip transaction pairs that clearly couldn't be attacks

Read-Write Conflict Check

Same Sender Check

## Vulnerability Localization

After finding an attack, identify which parts of the contract code are vulnerable.

Track how data changed by the attacker flows into the victim’s transaction.

Example: If attacker changes a contract’s balance, and victim’s function reads that balance to make decisions, that’s tainted.

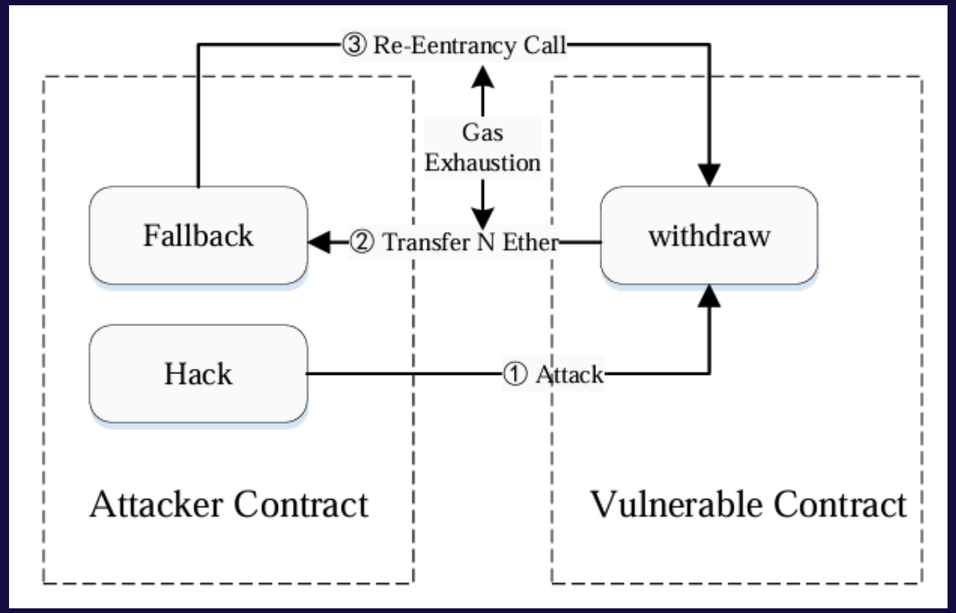
Instead of marking all the code as vulnerable, they only marked the tainted part.

**Paper 2**

**Reentrancy Vulnerabilities in Smart Contract Based on CPN**

# What is Reentrancy

Reentrancy is a bug in smart contracts where an attacker can call a function repeatedly before it finishes, allowing them to steal funds or manipulate the contract by exploiting this delay in execution.



# What tool did they use?

**They chose CPN because:**

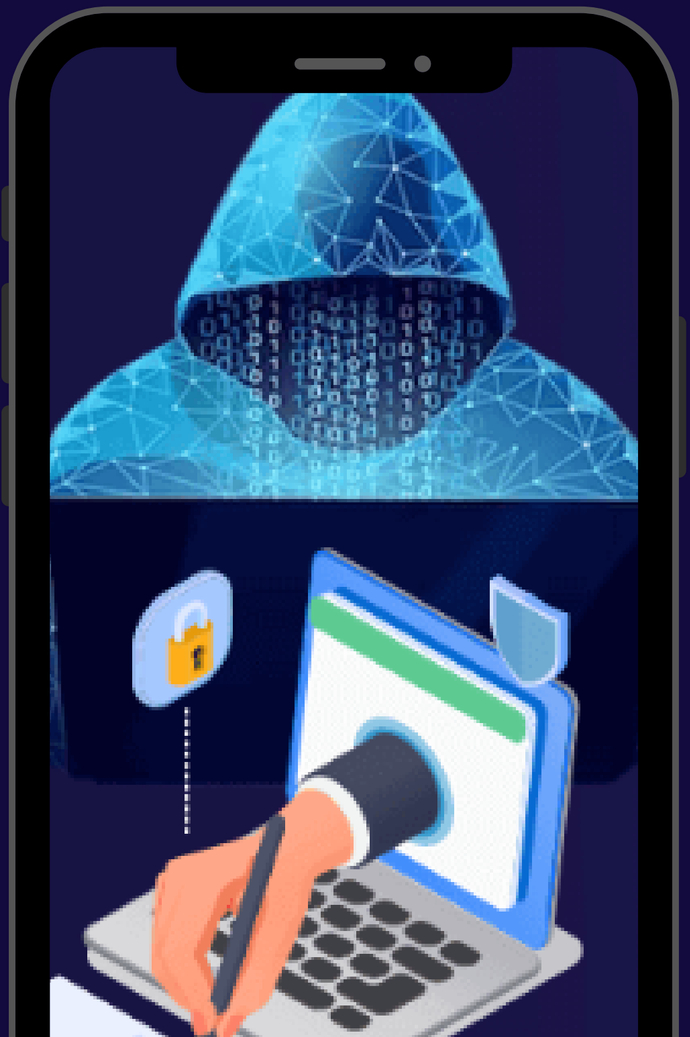
- It shows both the data flow and control flow — kind of like showing what the smart contract is doing AND thinking!
- It helps simulate the smart contract and check for bad paths — like testing if the candy jar can be stolen from before it locks.

**Paper 3**

**Reentrancy Vulnerabilities in Complex Smart Contracts on SliSE**

# Reentrancy

Reentrancy is a bug in smart contracts where an attacker can call a function repeatedly before it finishes, allowing them to steal funds or manipulate the contract by exploiting this delay in execution.



# What tool did they use?

## Paper 3

They used a tool called SliSE to find Reentrancy bugs in smart contracts.
It works in two steps: first, it cuts out the important parts of the code (called slicing), then it checks if a hacker can really break it using smart testing (symbolic execution).

# Conclusion

| Paper Title | Method Used | Explanation |
| --- | --- | --- |
| Combatting Front-Running in Smart Contracts | Attack Mining + Dynamic Taint Analysis | Mines real blockchain attacks and uses taint analysis to precisely localize front-running vulnerabilities. |
| Formal Analysis of Reentrancy Vulnerabilities in Smart Contract Based on CPN | Colored Petri Nets (CPN) | Applies CPN-based formal modeling and simulation to detect reentrancy via control and data flow analysis. |
| Efficiently Detecting Reentrancy Vulnerabilities in Complex Smart Contracts | SliSE (Program Slicing + Symbolic Execution) | Detects reentrancy in complex contracts through slicing and symbolic execution of inter-contract dependencies. |


Thank You