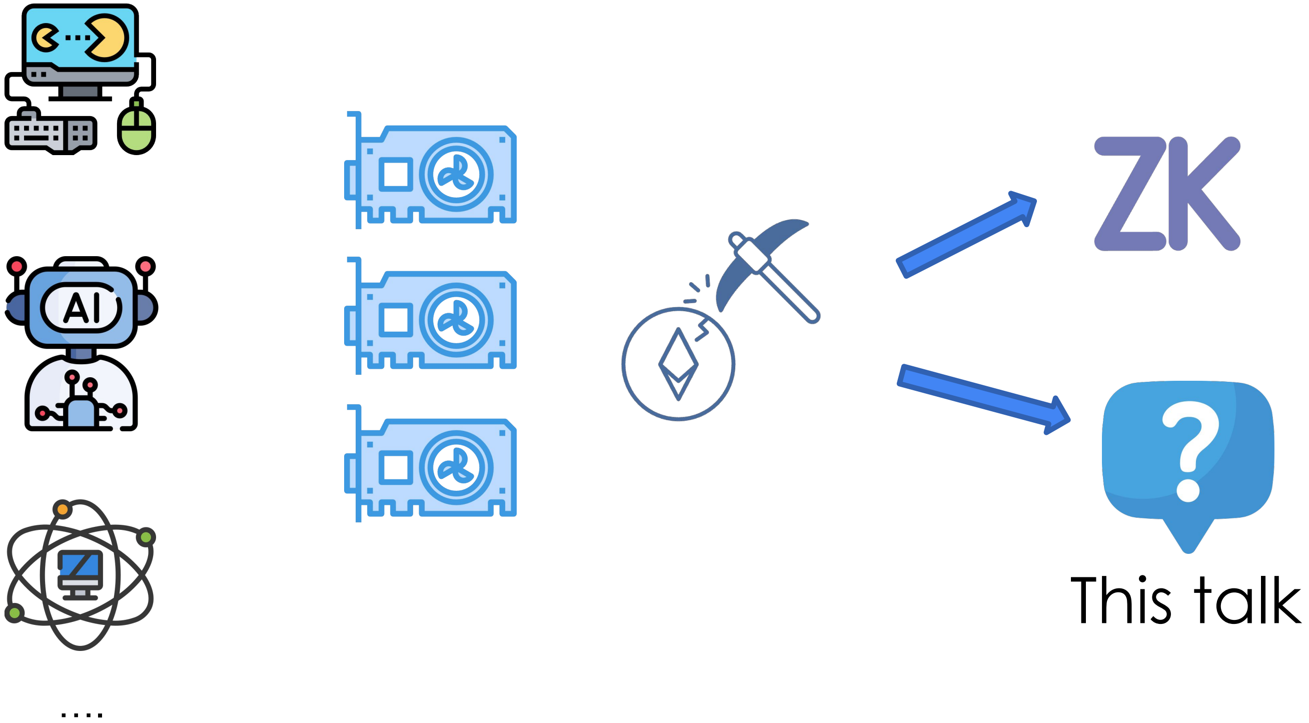
CuEVM: GPU-Accelerated EVM for Security and Beyond

Minh Ho

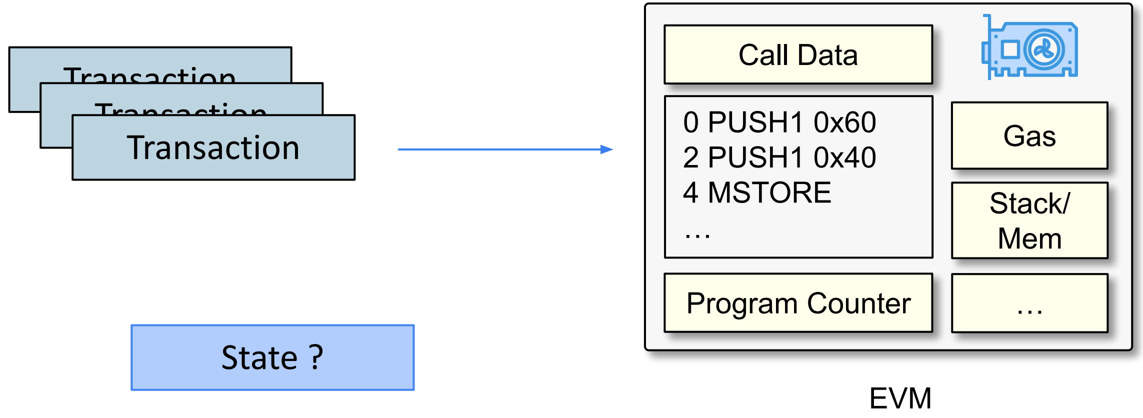
National University of Singapore


AI
....
ZK
This talk

Running Transactions in Parallel

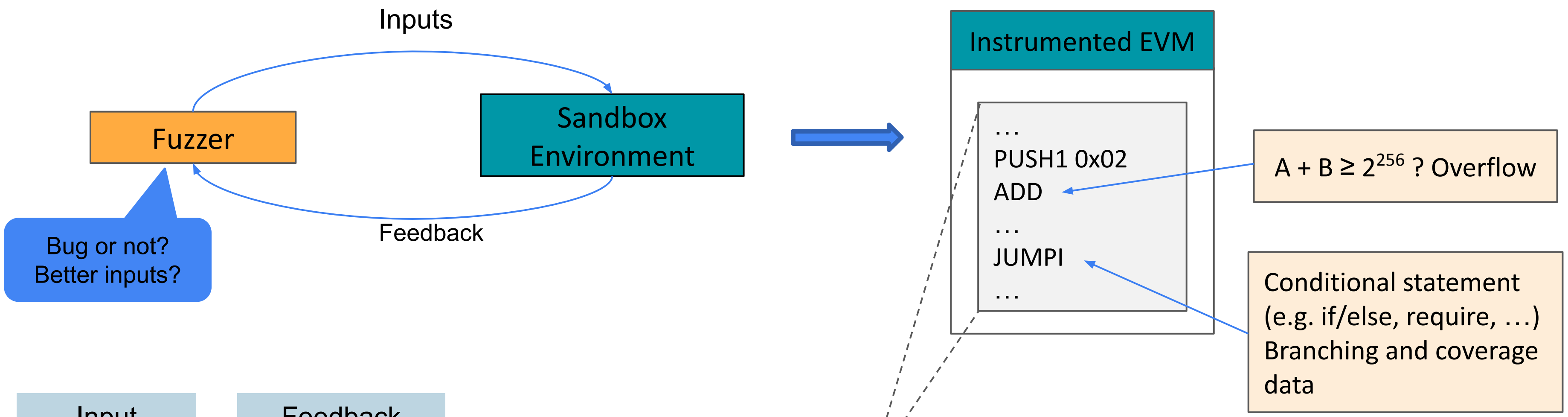
But why ?
How?

GPU – Massive Parallel Programming
Thousands of threads ⇔ thousands EVM instances



First: the niche fuzzing use case
Then: other potential use cases emerged

Why: Securing Smart Contracts

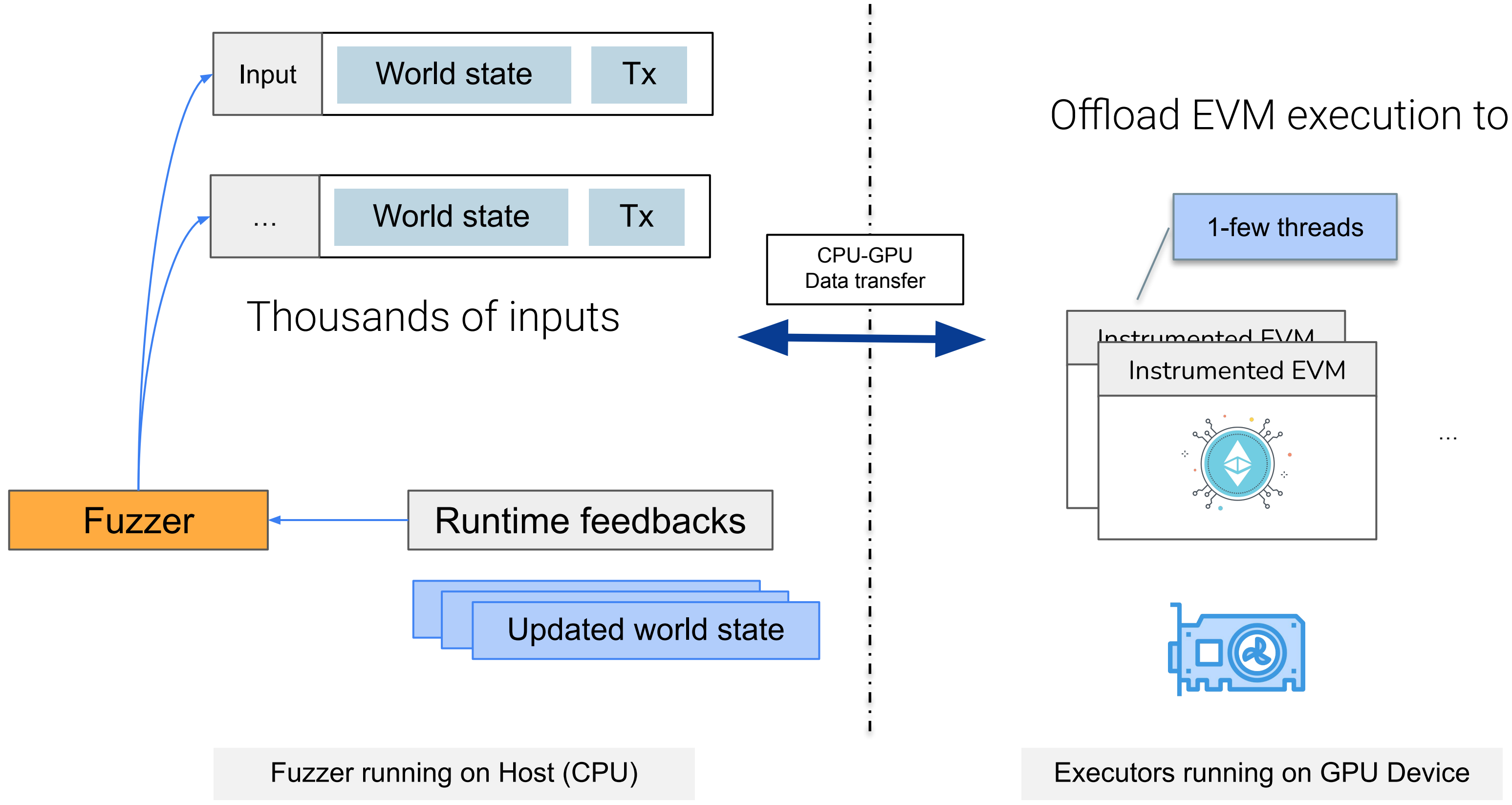

Inputs
Fuzzer
Sandbox Environment
Feedback
Bug or not? Better inputs?
Instrumented EVM
…
PUSH1 0x02
ADD
…
JUMPI
…
A + B ≥ 2^{256} ? Overflow
Conditional statement (e.g. if/else, require, …) Branching and coverage data

Input
State Transaction …

Feedback
Simplified traces Branching data New state …

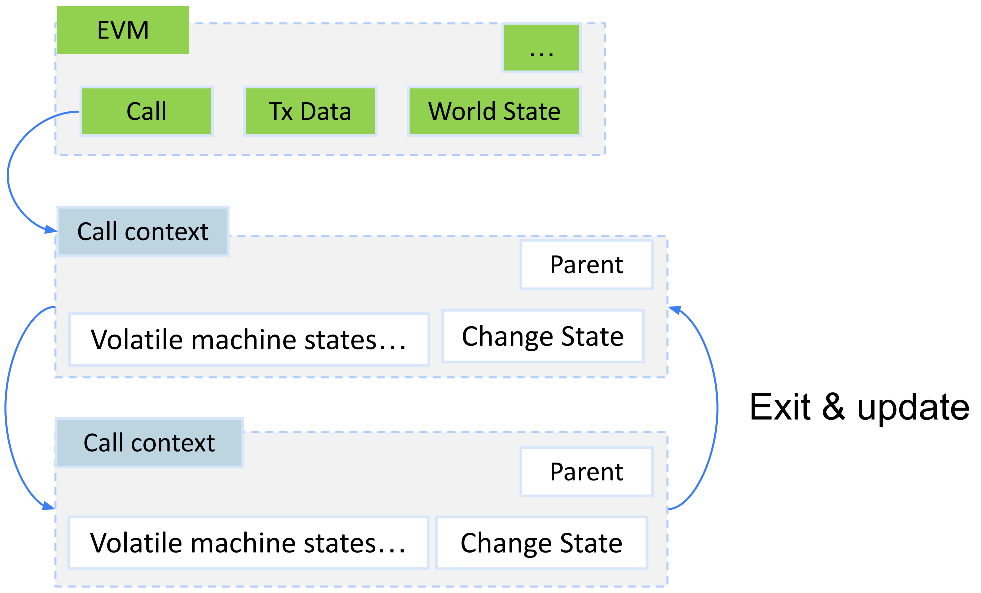
```
...
contract MyToken is ERC20 {
    ...
    // BUG
}
```

Why: Securing Smart Contracts

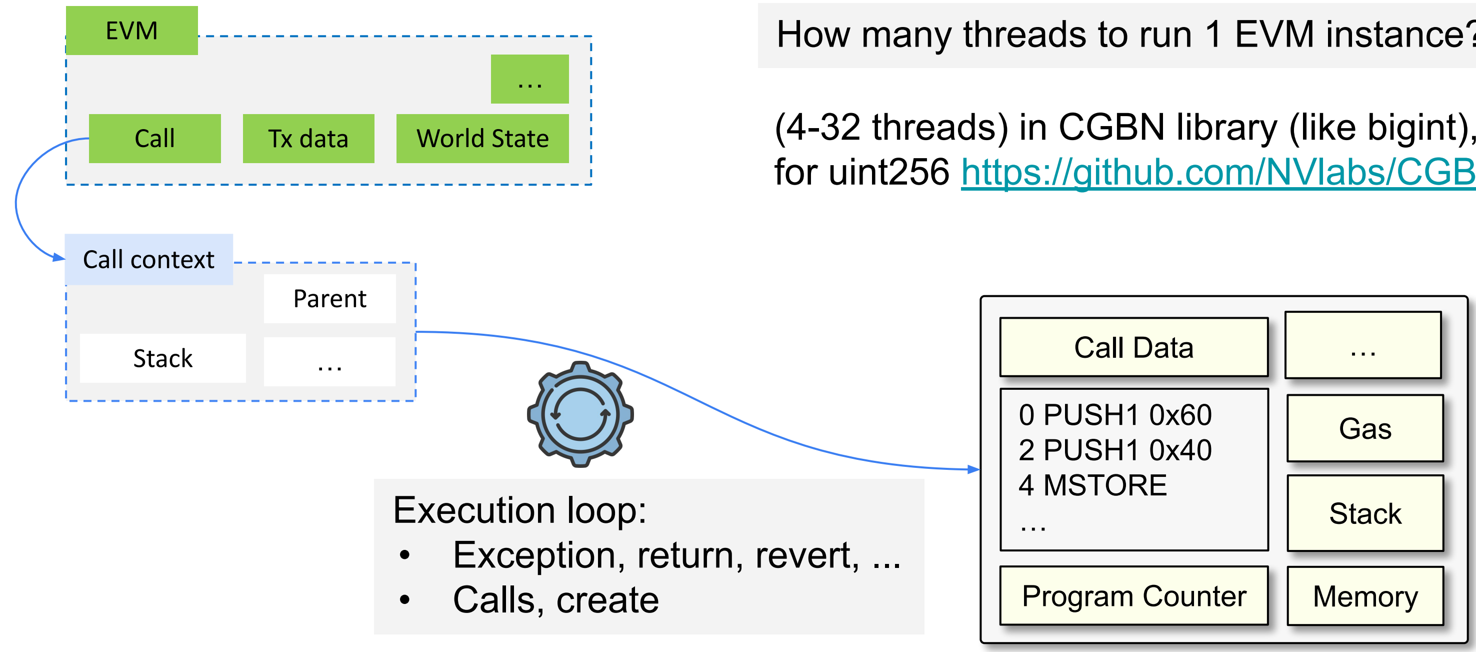


Mapping the EVM to GPU Threads

EVM instance ⇔ GPU thread(s)



Mapping the EVM to GPU Threads



How many threads to run 1 EVM instance?

(4-32 threads) in CGBN library (like bigint), configured for uint256 https://github.com/NVlabs/CGBN

Execution loop:

- Exception, return, revert, ...
- Calls, create

Ensure Correctness

EIP-3155: EVM trace specification
A JSON format for EVM traces

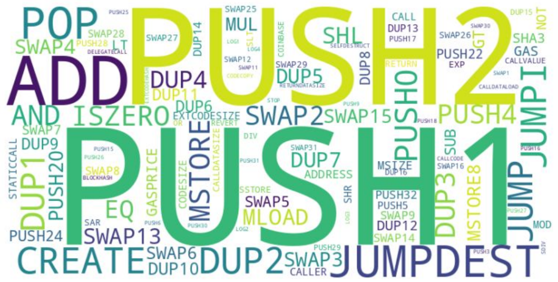
https://eips.ethereum.org/EIPS/eip-3155

Goevmlab integration and comparison with geth

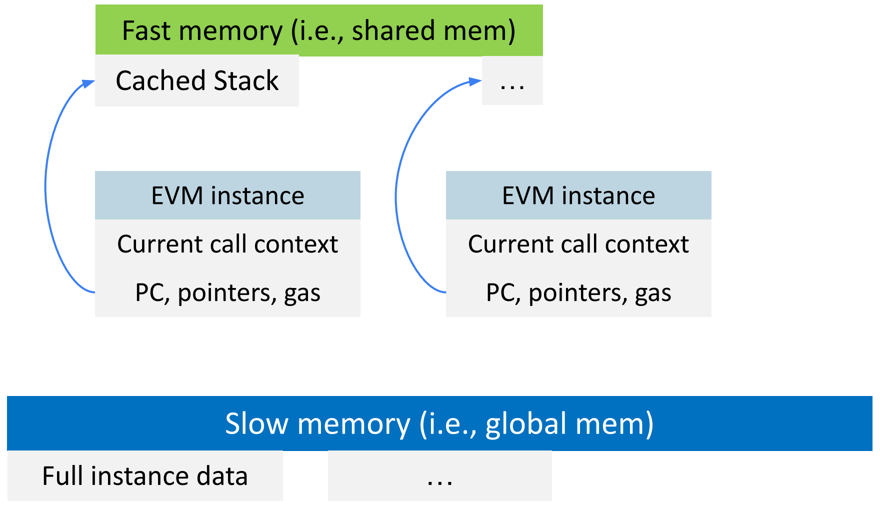
- ➔ Compare line by line in the trace
- ➔ We passed all tests in functional folders (VMTests, SystemOperations, Memory, Stack, ReturnData, Calls, Create, **PreCompiledContracts, ZeroKnowledge**)
- ➔ Rare corner cases remains (mainly gas difference), some resurface when optimizing

Preliminary Optimization

Stats of opcodes in 60 random historical blocks 2024



Median:
Max_stack_size: 22
Max_memory_size: 160
Max_pc: 8582



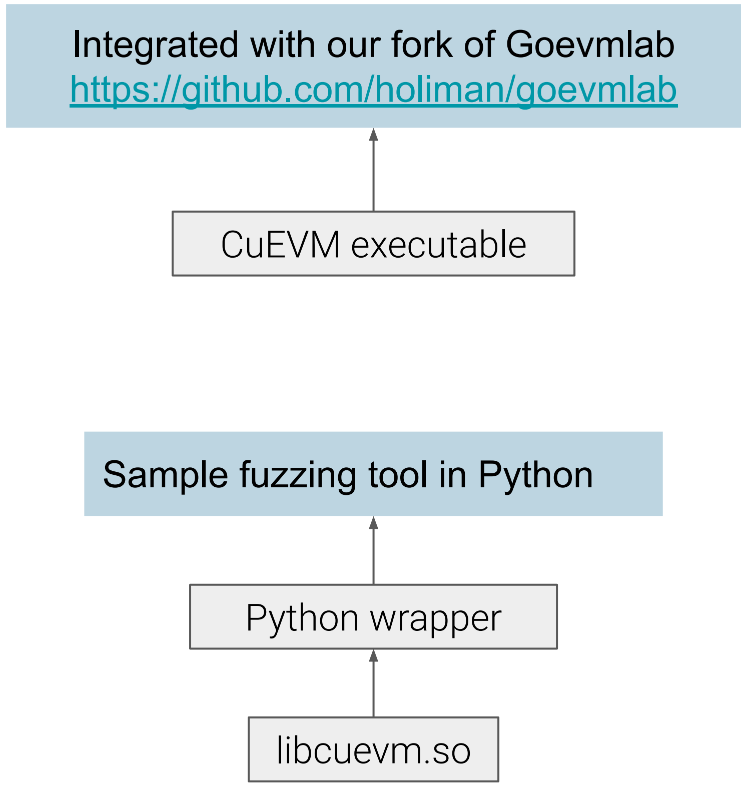
Small fixed-sized stack for each instance in faster memory

Current Implementation

- ➔ Shanghai and before
- ➔ Executable: output EIP-3155 JSON trace
- ➔ Two versions : CPU / GPU
- ➔ Shared library .so, for interacting from Python

Performance:

- Depending on the workload
- Preliminary testing 10x+ faster than our CPU version (deprecated)
- Requires more comprehensive benchmarking


Integrated with our fork of Goevmlab
https://github.com/holiman/goevmlab
CuEVM executable
Sample fuzzing tool in Python
Python wrapper
libcuevm.so

Sample run – GPU Fuzzing in Action

Google Colab demo - free (slow) GPUs

```
function bug_selfdestruct(uint input) public {
    require(input == 4567);
    selfdestruct(msg.sender);
}
function bug_unauthorized_send(uint input) public {
    if (input + 5678 == 45678) msg.sender.transfer(1 ether);
}
function test_branch(uint input) public  {require(input == 1000000); }
```

```
----------------------------------------
🚨 Bug Detected: Leaking Ether PC: 517, Line: 14
Function: bug_unauthorized_send
Inputs: [40000]
Source code:
        msg.sender.transfer(1 ether)

----------------------------------------
🚨 Bug Detected: selfdestruct PC: 445, Line: 8
Function: bug_selfdestruct
Inputs: [4567]
Source code:
        selfdestruct(msg.sender)
```

```
...
Branch 482,487 :
Seed(function='test branch', inputs=[ 64841], distance=935159)
Source code: require(input ==  1000000)
...
```

Current release: bottleneck in CPU fuzzing logic

Experimental removal of complex logic, does not conform with yellow paper:

- ~65k TPS on Nvidia L4 and improving
- Several hundred thousand TPS is feasible on multiple high-end GPUs ?!

Beyond Fuzzing

Parallelize transactions for Ethereum execution client?

Not very practical at the moment

- Need more transactions per block (thousands) and greater homogeneity

Client supports:

- Fetching a subset of relevant world state (memory constraints)
- Ensure it's safe & correct to run in parallel (deterministic + serializable)

Transaction simulation platform
Simulate similar txs on isolated states (e.g., simulate swap results)

Layer 2
Extremely high TPS if they can run safely in parallel

Team and Collaborators

Minh Ho, Chen Li
(Singapore Blockchain Innovation Programme, National University of Singapore)

Stefan-Dan Ciocîrlan
(Politehnica University of Bucharest, Romania)

Thanks to **Fredrik Svantes** and the **Ethereum Foundation** for advices and academic grant support

About us

Open-source at

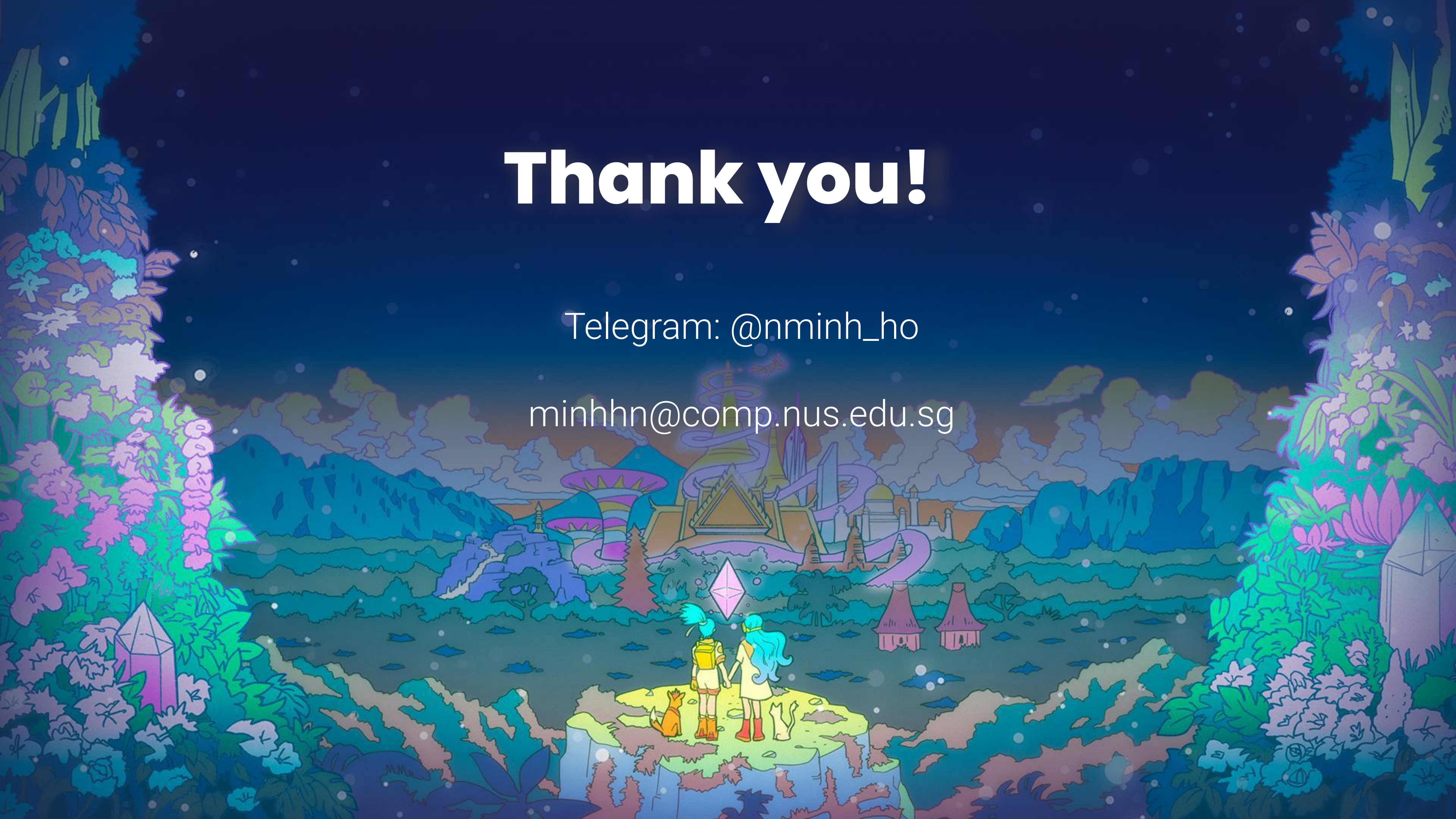
https://github.com/sbip-sg/CuEVM

Veratz Security

Singapore Blockchain Innovation Programme

NUS National University of Singapore | Computing

https://sbip.sg/



Thank you!

Telegram: @nminh_ho

minhhn@comp.nus.edu.sg