Curvance Internship

Learning Objective#2

Learn the Best and Latest Tools for Testing

@gas_limit

9th Oct to 11th Nov, 2023

Introduction

Weekly Goals & Performance

Date	Objectives	Summary
Oct 9 to Oct 15	Explore Foundry development environment	Completed most of the Foundry Book, participated in ETHGlobal Online hackathon
Oct 15 to Oct 22, 2023	Finish up Foundry & Explore Slither static analysis tool	Finished reading the Foundry Book, Began watching Owen Thurn’s new free security course, conducted slither static analysis
Oct 23 to Oct 29, 2023	Explore Olympix static analysis tool, begin Owen Thurm’s security course	Finished exploring olympix, finished 5 hours of the bootcamp, and was instructed to build a perpetual exchange over the next week
Oct 30 to Nov 5, 2023	Build a Perpetual Exchange	Finished Perpetual Exchange
Nov 6 to Nov 11, 2023	Explore Pyrometer static analysis tool	Completed pyrometer exploration and made necessary changes to contracts

Foundry Testing

Foundry is a comprehensive smart contract development and testing environment offering a suite of tools to streamline the process. It comprises a development toolchain, a testing framework called Forge, and CLI tools like Cast for interacting with deployed smart contracts. With features like inbuilt-fuzzing for Solidity scripting, assertions, and gas cost snapshots, it facilitates thorough testing. Additionally, Foundry supports deployment to various blockchain networks and provides a local node environment through Anvil for local testing. Its modular design, coupled with a user-friendly interface and extensive documentation, makes it a robust and accessible platform for developers venturing into smart contract development and deployment

Evaluating the functionality of expected failures using the "testFail" prefix.
Verifying the emission of transmissions through testing.
Conducting tests on an authorized counter contract.
Assessing whether an unauthorized user can perform an increment operation.

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.13;

import {Test, console2} from "forge-std/Test.sol";
import {AuthorizedCounter} from "../src/AuthorizedCounter.sol";

contract AuthorizedCounterTest is Test {
AuthorizedCounter public counter;
event CountChanged(uint256 newCount);

function setUp() public {
counter = new AuthorizedCounter();
}

// called by owner
function test_OwnerIncrement() public {
counter.increment();
}
// called by non-owner
function testFail_NonOwnerIncrement() public {
vm.prank(address(0));
counter.increment();
}
// test for correct event emission
function test_ExpectEmittedEvent() public {
vm.expectEmit(true, false, false, false);
uint256 count = counter.getCount();
emit CountChanged(count);
counter.increment();
}


}
```

Employing the "hoax" function
to restrict all transactions to be executed exclusively by specified users following their declaration.

```
function setUp() public {
address alice = address(0x123);
hoax(alice, 100 ether);
counter = new AuthorizedCounter();
}
```

```
// called by non-owner
function testFail_NonOwnerIncrement() public {
counter.increment();
}
```

Investigating the functionality of expected failures using the "expectRevert" method
(though less intuitive but possible).

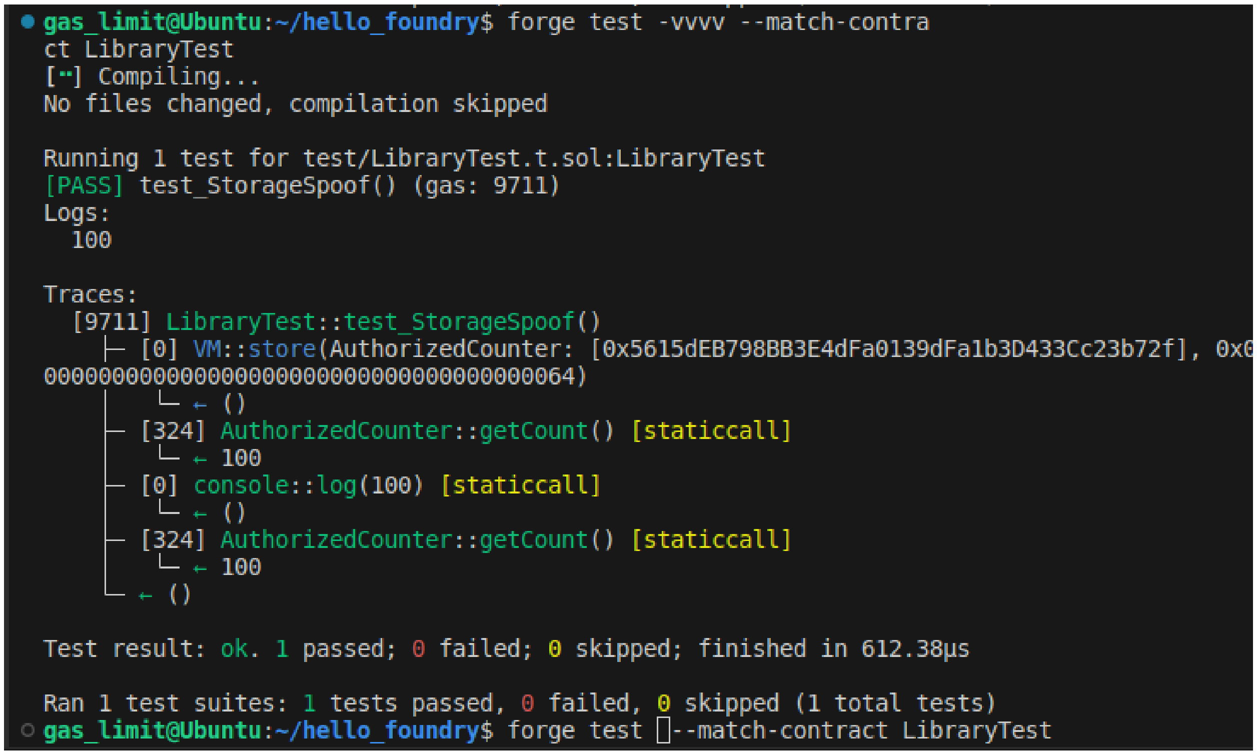
```
// called by non-owner
function test_NonOwnerIncrement() public {
```

```
vm.expectRevert();
counter.increment();
}
```

Experimenting with storage manipulation.

Testing the assertEq() function.

```
contract LibraryTest is Test {
AuthorizedCounter public AuthorizedCounter_;

function setUp() public {
AuthorizedCounter_ = new AuthorizedCounter();
}

function test_StorageSpoof() public {
vm.store(
address(AuthorizedCounter_),
bytes32(uint256(1)),
bytes32(uint256(100))
);
console2.log(AuthorizedCounter_.getCount());
assertEq((AuthorizedCounter_.getCount()), uint256(100));
}


}
```



```
● gas_limit@Ubuntu:~/hello_foundry$ forge test -vvvv --match-contra
  ct LibraryTest
  [⠒] Compiling...
  No files changed, compilation skipped

  Running 1 test for test/LibraryTest.t.sol:LibraryTest
  [PASS] test_StorageSpoof() (gas: 9711)
  Logs:
    100

  Traces:
    [9711] LibraryTest::test_StorageSpoof()
      ├─ [0] VM::store(AuthorizedCounter: [0x5615dEB798BB3E4dFa0139dFa1b3D433Cc23b72f], 0x0
  000000000000000000000000000000000000064)
      │   └─ ← ()
      ├─ [324] AuthorizedCounter::getCount() [staticcall]
      │   └─ ← 100
      ├─ [0] console::log(100) [staticcall]
      │   └─ ← ()
      ├─ [324] AuthorizedCounter::getCount() [staticcall]
      │   └─ ← 100
      └─ ← ()

  Test result: ok. 1 passed; 0 failed; 0 skipped; finished in 612.38µs

  Ran 1 test suites: 1 tests passed, 0 failed, 0 skipped (1 total tests)
○ gas_limit@Ubuntu:~/hello_foundry$ forge test --match-contract LibraryTest
```

Analyzing test results and enhancing verbosity with the '-vv' flag
to include logs from 'console2.log'.

Employing the '-vvvv' option to enable tracing in all tests.

Evaluating a larger contract, 'UniswapV2Liquidity,' designed for transferring user funds to this contract and subsequently facilitating deposits and withdrawals within the UniswapV2 Protocol.

This is the contract to test:

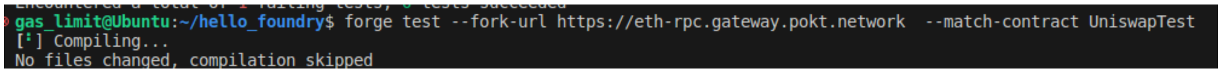
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol";

contract UniswapV2AddLiquidity {
address private constant FACTORY =
0x5C69bEe701ef814a2B6a3EDD4B1652CB9cc5aA6f;
address private constant ROUTER =
0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D;
address private constant WETH =
0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2;
```

```
address private constant USDT =
0xdAC17F958D2ee523a2206206994597C13D831ec7;

function addLiquidity(
address _tokenA,
address _tokenB,
uint _amountA,
uint _amountB
) external {
safeTransferFrom(IERC20(_tokenA), msg.sender, address(this), _amountA);
safeTransferFrom(IERC20(_tokenB), msg.sender, address(this), _amountB);

safeApprove(IERC20(_tokenA), ROUTER, _amountA);
safeApprove(IERC20(_tokenB), ROUTER, _amountB);

(uint amountA, uint amountB, uint liquidity) = IUniswapV2Router(ROUTER)
.addLiquidity(
_tokenA,
_tokenB,
_amountA,
_amountB,
1,
1,
address(this),
block.timestamp
);
}

function removeLiquidity(address _tokenA, address _tokenB) external {
address pair = IUniswapV2Factory(FACTORY).getPair(_tokenA, _tokenB);

uint liquidity = IERC20(pair).balanceOf(address(this));
safeApprove(IERC20(pair), ROUTER, liquidity);

(uint amountA, uint amountB) = IUniswapV2Router(ROUTER).removeLiquidity(
_tokenA,
_tokenB,
liquidity,
1,
1,
address(this),
block.timestamp
);
}

/**
* @dev The transferFrom function may or may not return a bool.
* The ERC-20 spec returns a bool, but some tokens don't follow the spec.
* Need to check if data is empty or true.
*/
function safeTransferFrom(
IERC20 token,
address sender,
address recipient,
uint amount
) internal {
(bool success, bytes memory returnData) = address(token).call(
abi.encodeCall(IERC20.transferFrom, (sender, recipient, amount))
);
require(
success && (returnData.length == 0 || abi.decode(returnData, (bool))),
"Transfer from fail"
);
}

/**
* @dev The approve function may or may not return a bool.
* The ERC-20 spec returns a bool, but some tokens don't follow the spec.
* Need to check if data is empty or true.
*/
function safeApprove(IERC20 token, address spender, uint amount) internal
{
(bool success, bytes memory returnData) = address(token).call(
abi.encodeCall(IERC20.approve, (spender, amount))
```

```
);
require(
success && (returnData.length == 0 || abi.decode(returnData, (bool))),
"Approve fail"
);
}
}

interface IUniswapV2Router {
function addLiquidity(
address tokenA,
address tokenB,
uint amountADesired,
uint amountBDesired,
uint amountAMin,
uint amountBMin,
address to,
uint deadline
) external returns (uint amountA, uint amountB, uint liquidity);

function removeLiquidity(
address tokenA,
address tokenB,
uint liquidity,
uint amountAMin,
uint amountBMin,
address to,
uint deadline
) external returns (uint amountA, uint amountB);
}

interface IUniswapV2Factory {
function getPair(address token0, address token1) external view returns
(address);
}
```

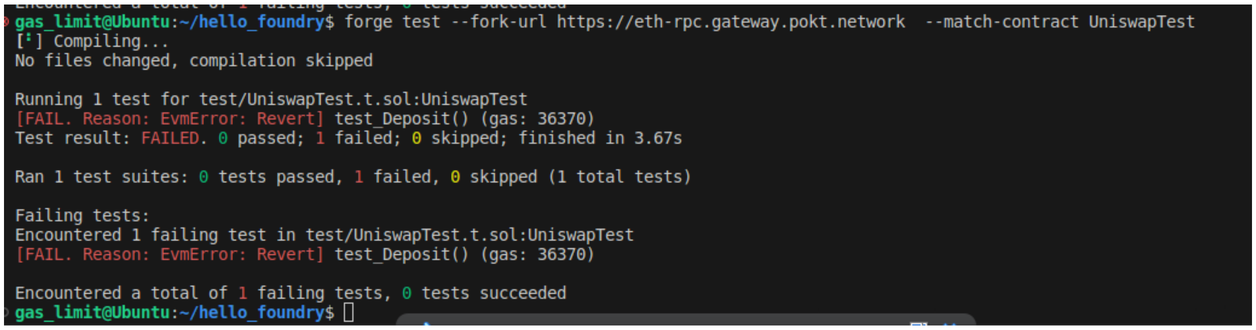
Transitioning to the use of an Ethereum mainnet fork for testing,

as opposed to local testing, moving forward.



```
gas_limit@Ubuntu:~/hello_foundry$ forge test --fork-url https://eth-rpc.gateway.pokt.network  --match-contract UniswapTest
[⠃] Compiling...
No files changed, compilation skipped
```

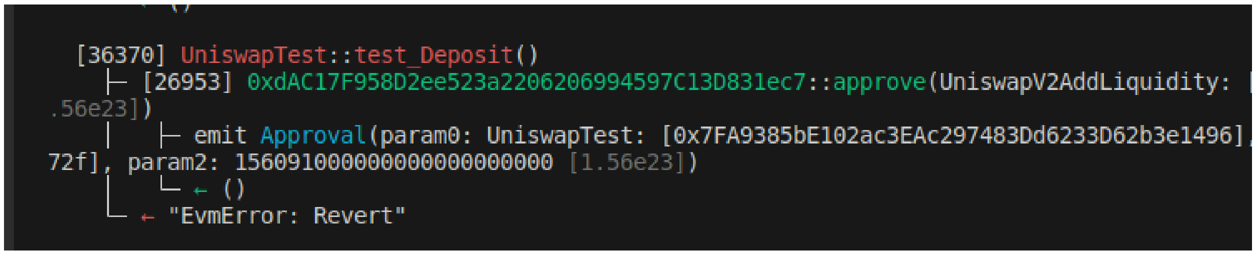
I encountered difficulties when attempting to test the 'approve()' function



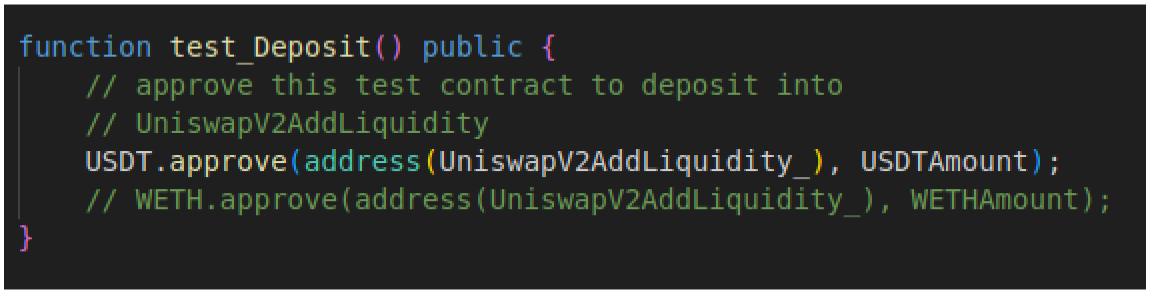
```
gas_limit@Ubuntu:~/hello_foundry$ forge test --fork-url https://eth-rpc.gateway.pokt.network  --match-contract UniswapTest
[⠃] Compiling...
No files changed, compilation skipped

Running 1 test for test/UniswapTest.t.sol:UniswapTest
[FAIL. Reason: EvmError: Revert] test_Deposit() (gas: 36370)
Test result: FAILED. 0 passed; 1 failed; 0 skipped; finished in 3.67s

Ran 1 test suites: 0 tests passed, 1 failed, 0 skipped (1 total tests)

Failing tests:
Encountered 1 failing test in test/UniswapTest.t.sol:UniswapTest
[FAIL. Reason: EvmError: Revert] test_Deposit() (gas: 36370)

Encountered a total of 1 failing tests, 0 tests succeeded
gas_limit@Ubuntu:~/hello_foundry$
```

Employing the '-vvvv' flag

to conduct more thorough diagnostic testing below.

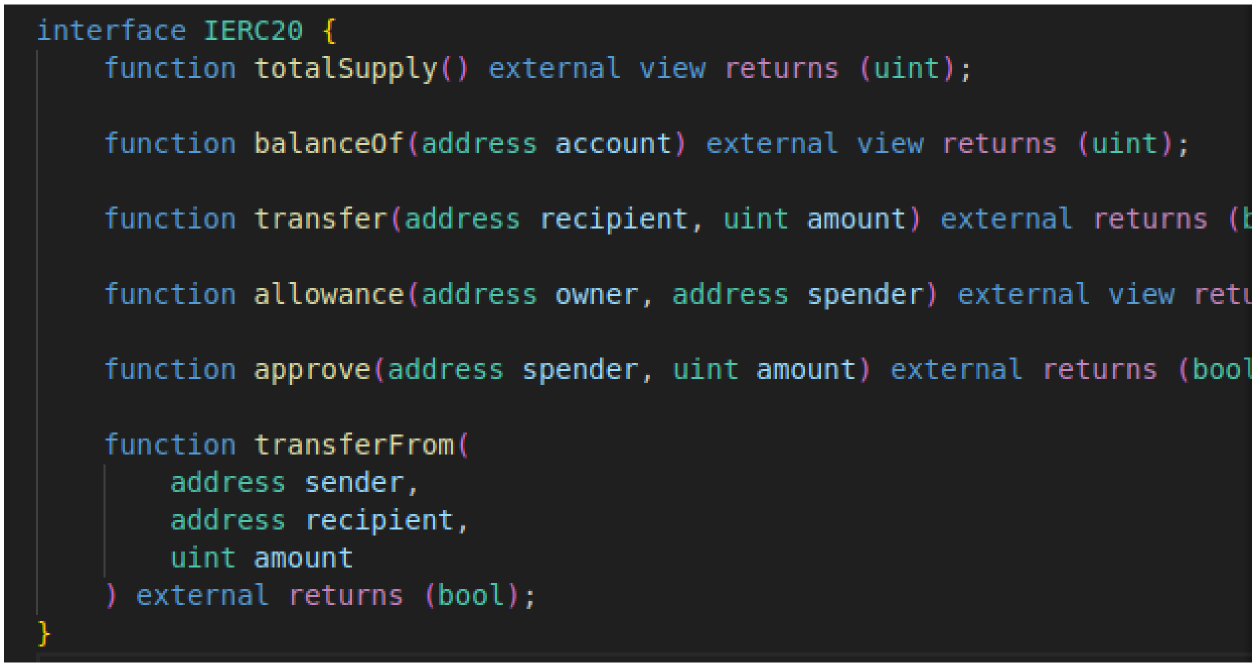


```
  [36370] UniswapTest::test_Deposit()
    ├─ [26953] 0xdAC17F958D2ee523a2206206994597C13D831ec7::approve(UniswapV2AddLiquidity: [
 .56e23])
    │   ├─ emit Approval(param0: UniswapTest: [0x7FA9385bE102ac3EAc297483Dd6233D62b3e1496],
 72f], param2: 156091000000000000000000 [1.56e23])
    │   └─ ← ()
    └─ ← "EvmError: Revert"
```



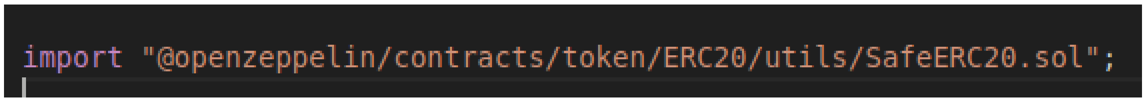
```
function test_Deposit() public {
    // approve this test contract to deposit into
    // UniswapV2AddLiquidity
    USDT.approve(address(UniswapV2AddLiquidity_), USDTAmount);
    // WETH.approve(address(UniswapV2AddLiquidity_), WETHAmount);
}
```

After consulting with the developer community, it was recommended that I utilize the SafeERC20 Library instead of making direct calls to the IERC20 interface.

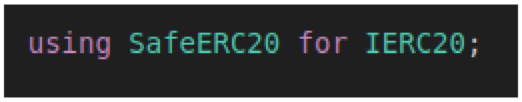


```
interface IERC20 {
    function totalSupply() external view returns (uint);

    function balanceOf(address account) external view returns (uint);

    function transfer(address recipient, uint amount) external returns (b

    function allowance(address owner, address spender) external view retu

    function approve(address spender, uint amount) external returns (bool

    function transferFrom(
        address sender,
        address recipient,
        uint amount
    ) external returns (bool);
}
```

IERC20 interface ^



```
import "@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol";
```



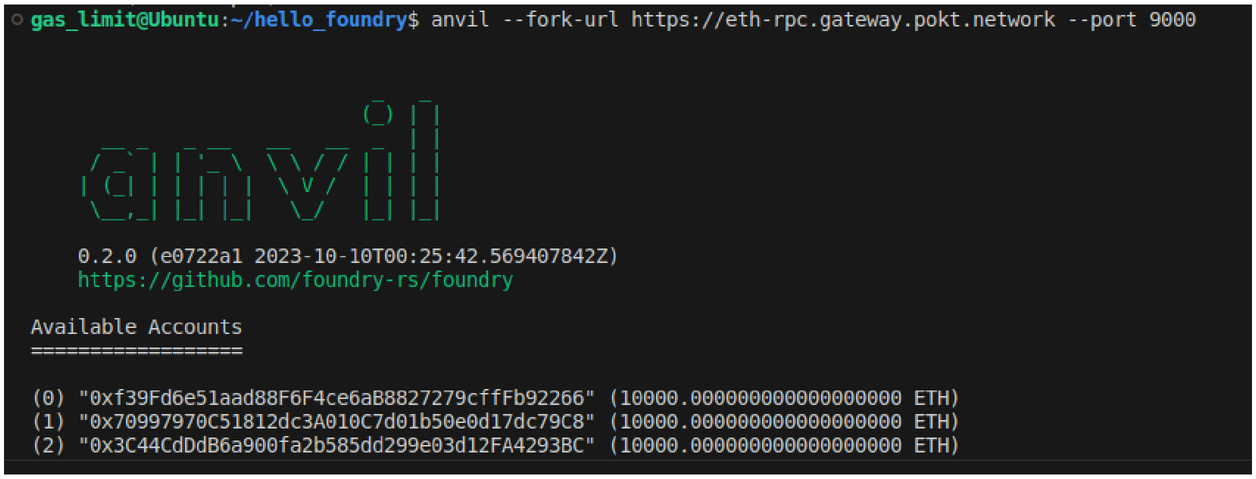
```
using SafeERC20 for IERC20;
```

Following multiple test runs, I encountered RPC URL-related errors due to timeouts, and I received advice to deploy a local fork using anvil.
With the command:

```
anvil --fork-url https://eth-rpc.gateway.pokt.network -port
9000
```

And I am using the usual testing command but using port9000 as the target fork blockchain using:

```
forge test --fork-url https://localhost:9000
```



```
gas_limit@Ubuntu:~/hello_foundry$ anvil --fork-url https://eth-rpc.gateway.pokt.network --port 9000


                             _   _
                            (_) | |
      __ _   _ __   __   __  _  | |
     / _` | | '_ \  \ \ / / | | | |
    | (_| | | | | |  \ V /  | | | |
     \__,_| |_| |_|   \_/   |_| |_|

    0.2.0 (e0722a1 2023-10-10T00:25:42.569407842Z)
    https://github.com/foundry-rs/foundry

Available Accounts
==================

(0) "0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266" (10000.000000000000000000 ETH)
(1) "0x70997970C51812dc3A010C7d01b50e0d17dc79C8" (10000.000000000000000000 ETH)
(2) "0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC" (10000.000000000000000000 ETH)
```

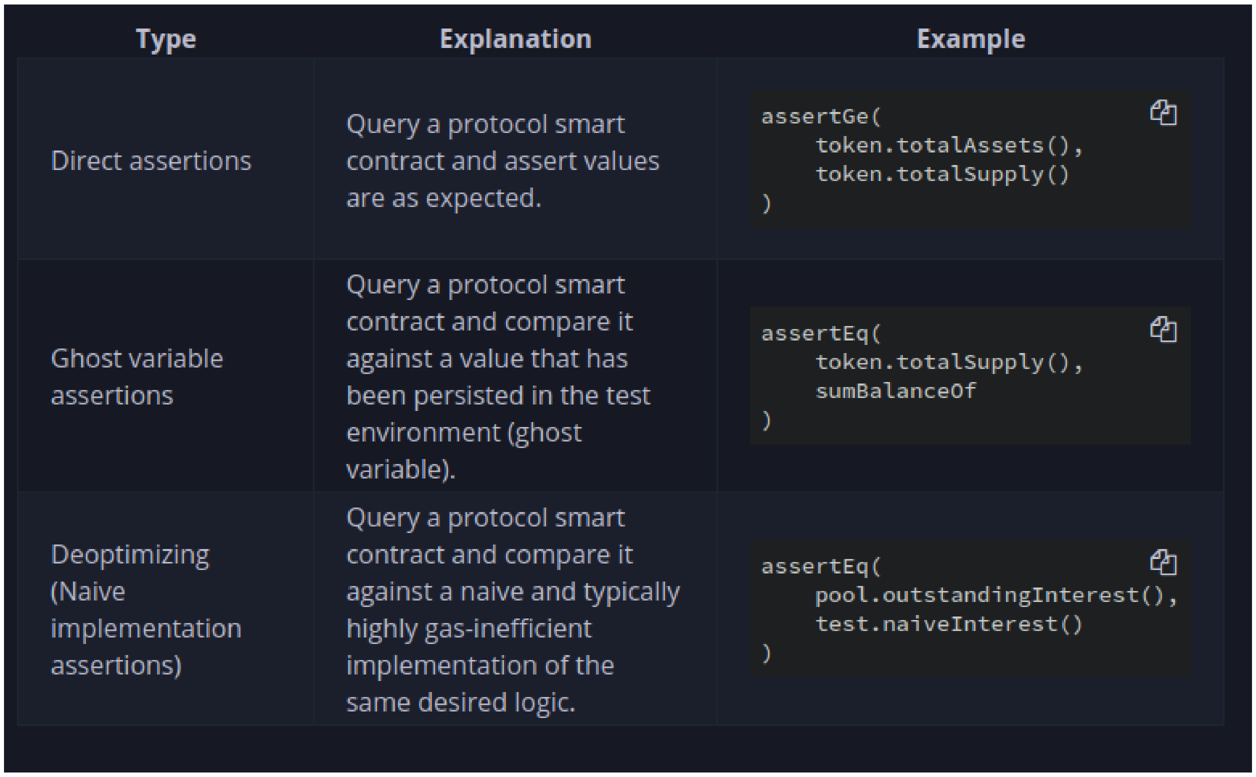
Conducting fuzz testing.

The default setting tests 256 values, which can be adjusted by modifying the 'foundry.toml' configuration file.

```
import "forge-std/Test.sol";

contract Safe {
receive() external payable {}

function withdraw() external {
payable(msg.sender).transfer(address(this).balance);
}
}

contract SafeTest is Test {
Safe safe;

// Needed so the test contract itself can receive ether
// when withdrawing
receive() external payable {}

function setUp() public {
safe = new Safe();
}
function testFuzz_Withdraw(uint96 amount) public {
vm.assume(amount > 0.1 ether);
payable(address(safe)).transfer(amount);
uint256 preBalance = address(this).balance;
safe.withdraw();
uint256 postBalance = address(this).balance;
assertEq(preBalance + amount, postBalance);
}
}
```

Invariant testing

Employing the prefix 'invariant_exampleFunction()' for testing and invariants.

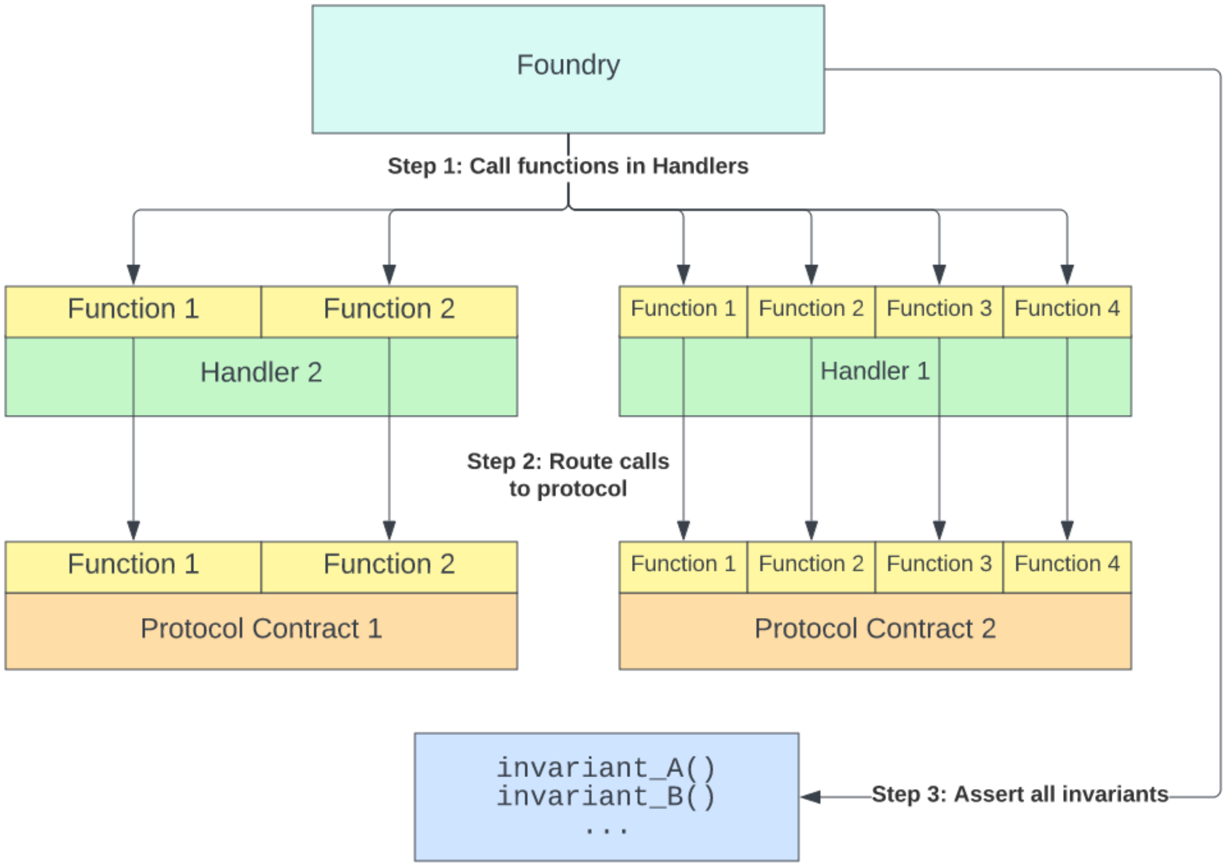


Type	Explanation	Example
Direct assertions	Query a protocol smart contract and assert values are as expected.	`assertGe( token.totalAssets(), token.totalSupply() )`
Ghost variable assertions	Query a protocol smart contract and compare it against a value that has been persisted in the test environment (ghost variable).	`assertEq( token.totalSupply(), sumBalanceOf )`
Deoptimizing (Naive implementation assertions)	Query a protocol smart contract and compare it against a naive and typically highly gas-inefficient implementation of the same desired logic.	`assertEq( pool.outstandingInterest(), test.naiveInterest() )`

Handler Contracts

A handler contract is used to test more complex protocols or contracts. It works as a wrapper contract that will be used to interact or make calls to our desired contract.

It is particularly necessary when the environment needs to be configured in a certain way (i.e. a constructor is called with certain parameters).



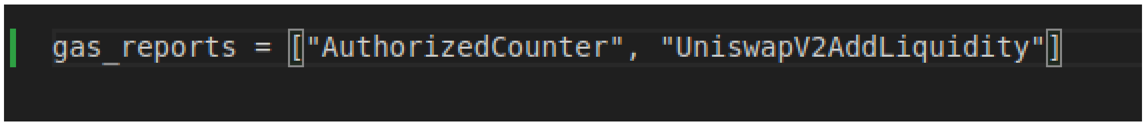
Differential Testing
Use mathematical operations from other languages and compare them to solidity operations
Python Implementation:
https://github.com/FrankieIsLost/gradual-dutch-auction/blob/master/analysis/compute_price.py

Video of python implementation

https://www.youtube.com/watch?v=WhZQhxOG124

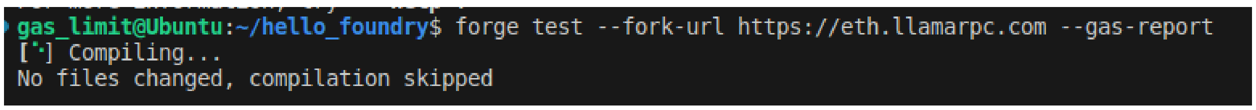
Gas Reports

Initially, I updated my 'foundry.toml' file and included the 'gas_reports' field.



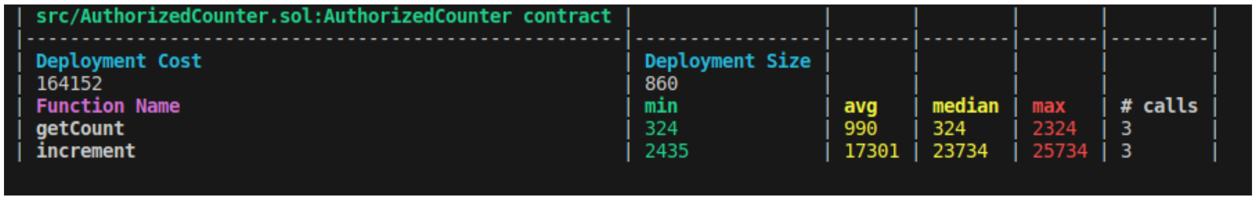
```
gas_reports = ["AuthorizedCounter", "UniswapV2AddLiquidity"]
```

During the forked mainnet test, I utilized the '--gas-report' flag.



```
gas_limit@Ubuntu:~/hello_foundry$ forge test --fork-url https://eth.llamarpc.com --gas-report
[⠊] Compiling...
No files changed, compilation skipped
```

The gas report of AuthorizedCounter



```
| src/AuthorizedCounter.sol:AuthorizedCounter contract |                 |       |        |       |         |
|------------------------------------------------------|-----------------|-------|--------|-------|---------|
| Deployment Cost                                      | Deployment Size |       |        |       |         |
| 164152                                               | 860             |       |        |       |         |
| Function Name                                        | min             | avg   | median | max   | # calls |
| getCount                                             | 324             | 990   | 324    | 2324  | 3       |
| increment                                            | 2435            | 17301 | 23734  | 25734 | 3       |
```

Deployment cost: 164,152 gas (0.000000000000164152 ether)

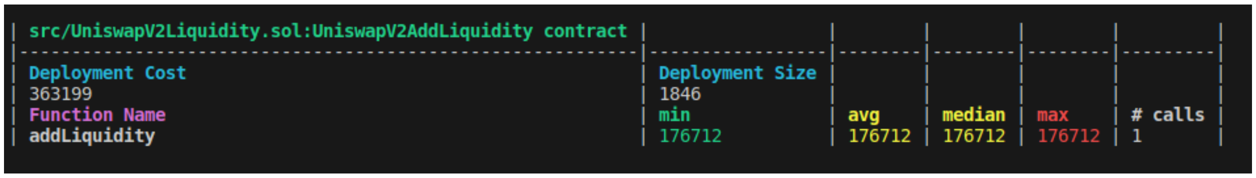
getCount function call:

Minimum cost: 324 gas average: 990 max: 2324

Increment function call:

Minimum cost: 2435 average: 17301 max: 25734

The gas report of UniswapV2AddLiquidity



```
| src/UniswapV2Liquidity.sol:UniswapV2AddLiquidity contract |                 |        |        |        |         |
|-----------------------------------------------------------|-----------------|--------|--------|--------|---------|
| Deployment Cost                                           | Deployment Size |        |        |        |         |
| 363199                                                    | 1846            |        |        |        |         |
| Function Name                                             | min             | avg    | median | max    | # calls |
| addLiquidity                                              | 176712          | 176712 | 176712 | 176712 | 1       |
```

Deployment cost: 363,199

addLiquidity function call:

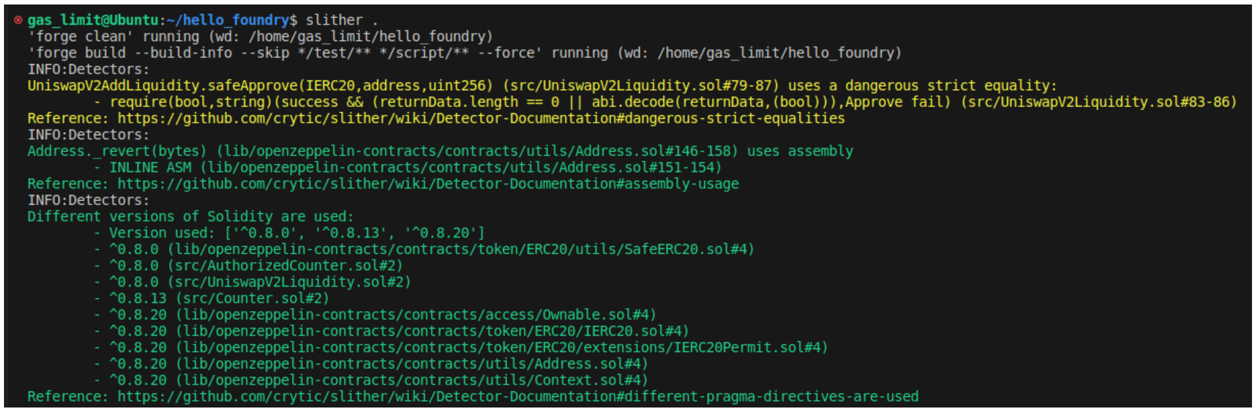
Minimum cost: 176712 average: 176712 (same) max: 176712

Slither Static Analysis

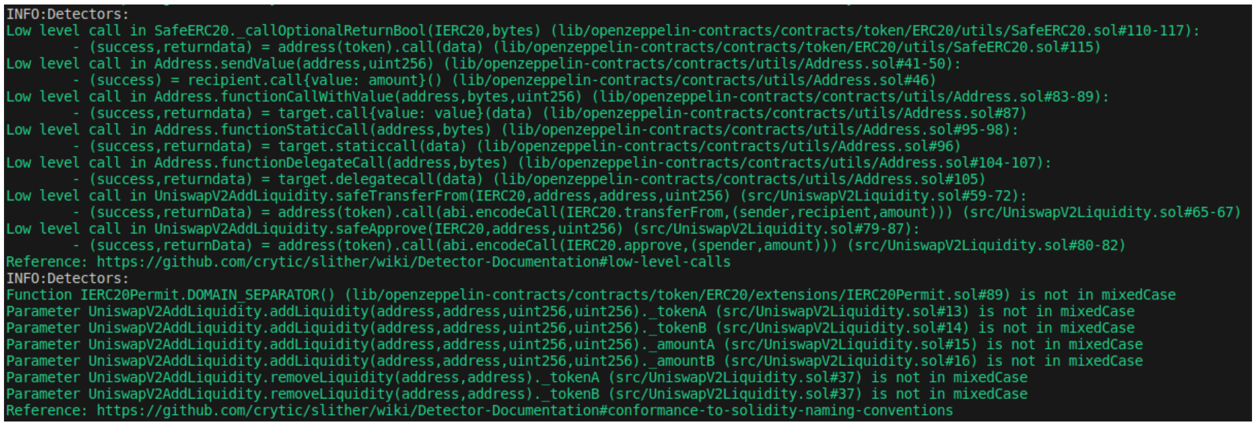
Slither is a Solidity & Vyper static analysis framework written in Python3. It runs a suite of vulnerability detectors, prints visual information about contract details, and provides an API to easily write custom analyses. Slither enables developers to find vulnerabilities, enhance their code comprehension, and quickly prototype custom analyses.

For this test, I am not trying to find out the accuracy of the tools findings, only explore its features. Testing on a more complex contract can be seen here

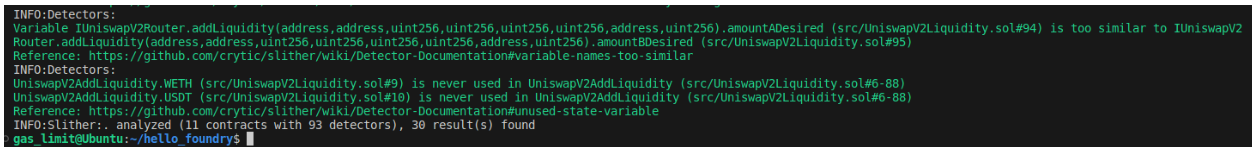
Slither general test using ‘slither .’



```
⊗ gas_limit@Ubuntu:~/hello_foundry$ slither .
  'forge clean' running (wd: /home/gas_limit/hello_foundry)
  'forge build --build-info --skip */test/** */script/** --force' running (wd: /home/gas_limit/hello_foundry)
  INFO:Detectors:
  UniswapV2AddLiquidity.safeApprove(IERC20,address,uint256) (src/UniswapV2Liquidity.sol#79-87) uses a dangerous strict equality:
          - require(bool,string)(success && (returnData.length == 0 || abi.decode(returnData,(bool))),Approve fail) (src/UniswapV2Liquidity.sol#83-86)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
  INFO:Detectors:
  Address._revert(bytes) (lib/openzeppelin-contracts/contracts/utils/Address.sol#146-158) uses assembly
          - INLINE ASM (lib/openzeppelin-contracts/contracts/utils/Address.sol#151-154)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
  INFO:Detectors:
  Different versions of Solidity are used:
          - Version used: ['^0.8.0', '^0.8.13', '^0.8.20']
          - ^0.8.0 (lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol#4)
          - ^0.8.0 (src/AuthorizedCounter.sol#2)
          - ^0.8.0 (src/UniswapV2Liquidity.sol#2)
          - ^0.8.13 (src/Counter.sol#2)
          - ^0.8.20 (lib/openzeppelin-contracts/contracts/access/Ownable.sol#4)
          - ^0.8.20 (lib/openzeppelin-contracts/contracts/token/ERC20/IERC20.sol#4)
          - ^0.8.20 (lib/openzeppelin-contracts/contracts/token/ERC20/extensions/IERC20Permit.sol#4)
          - ^0.8.20 (lib/openzeppelin-contracts/contracts/utils/Address.sol#4)
          - ^0.8.20 (lib/openzeppelin-contracts/contracts/utils/Context.sol#4)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
```

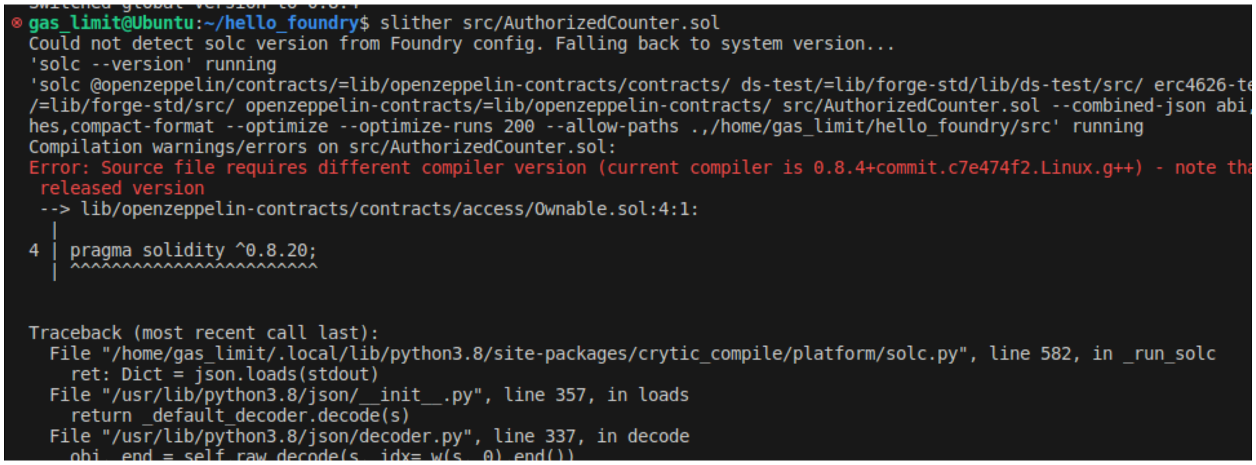


```
INFO:Detectors:
Low level call in SafeERC20._callOptionalReturnBool(IERC20,bytes) (lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol#110-117):
        - (success,returndata) = address(token).call(data) (lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol#115)
Low level call in Address.sendValue(address,uint256) (lib/openzeppelin-contracts/contracts/utils/Address.sol#41-50):
        - (success) = recipient.call{value: amount}() (lib/openzeppelin-contracts/contracts/utils/Address.sol#46)
Low level call in Address.functionCallWithValue(address,bytes,uint256) (lib/openzeppelin-contracts/contracts/utils/Address.sol#83-89):
        - (success,returndata) = target.call{value: value}(data) (lib/openzeppelin-contracts/contracts/utils/Address.sol#87)
Low level call in Address.functionStaticCall(address,bytes) (lib/openzeppelin-contracts/contracts/utils/Address.sol#95-98):
        - (success,returndata) = target.staticcall(data) (lib/openzeppelin-contracts/contracts/utils/Address.sol#96)
Low level call in Address.functionDelegateCall(address,bytes) (lib/openzeppelin-contracts/contracts/utils/Address.sol#104-107):
        - (success,returndata) = target.delegatecall(data) (lib/openzeppelin-contracts/contracts/utils/Address.sol#105)
Low level call in UniswapV2AddLiquidity.safeTransferFrom(IERC20,address,address,uint256) (src/UniswapV2Liquidity.sol#59-72):
        - (success,returnData) = address(token).call(abi.encodeCall(IERC20.transferFrom,(sender,recipient,amount))) (src/UniswapV2Liquidity.sol#65-67)
Low level call in UniswapV2AddLiquidity.safeApprove(IERC20,address,uint256) (src/UniswapV2Liquidity.sol#79-87):
        - (success,returnData) = address(token).call(abi.encodeCall(IERC20.approve,(spender,amount))) (src/UniswapV2Liquidity.sol#80-82)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IERC20Permit.DOMAIN_SEPARATOR() (lib/openzeppelin-contracts/contracts/token/ERC20/extensions/IERC20Permit.sol#89) is not in mixedCase
Parameter UniswapV2AddLiquidity.addLiquidity(address,address,uint256,uint256)._tokenA (src/UniswapV2Liquidity.sol#13) is not in mixedCase
Parameter UniswapV2AddLiquidity.addLiquidity(address,address,uint256,uint256)._tokenB (src/UniswapV2Liquidity.sol#14) is not in mixedCase
Parameter UniswapV2AddLiquidity.addLiquidity(address,address,uint256,uint256)._amountA (src/UniswapV2Liquidity.sol#15) is not in mixedCase
Parameter UniswapV2AddLiquidity.addLiquidity(address,address,uint256,uint256)._amountB (src/UniswapV2Liquidity.sol#16) is not in mixedCase
Parameter UniswapV2AddLiquidity.removeLiquidity(address,address)._tokenA (src/UniswapV2Liquidity.sol#37) is not in mixedCase
Parameter UniswapV2AddLiquidity.removeLiquidity(address,address)._tokenB (src/UniswapV2Liquidity.sol#37) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```



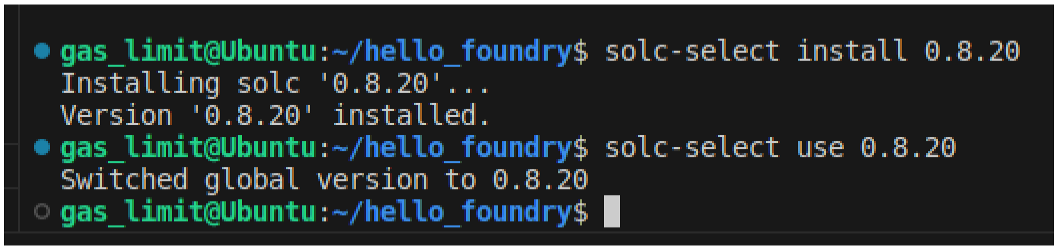
```
INFO:Detectors:
Variable IUniswapV2Router.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (src/UniswapV2Liquidity.sol#94) is too similar to IUniswapV2
Router.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (src/UniswapV2Liquidity.sol#95)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
UniswapV2AddLiquidity.WETH (src/UniswapV2Liquidity.sol#9) is never used in UniswapV2AddLiquidity (src/UniswapV2Liquidity.sol#6-88)
UniswapV2AddLiquidity.USDT (src/UniswapV2Liquidity.sol#10) is never used in UniswapV2AddLiquidity (src/UniswapV2Liquidity.sol#6-88)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
INFO:Slither:. analyzed (11 contracts with 93 detectors), 30 result(s) found
gas_limit@Ubuntu:~/hello_foundry$ 
```

Static analysis on a specific file



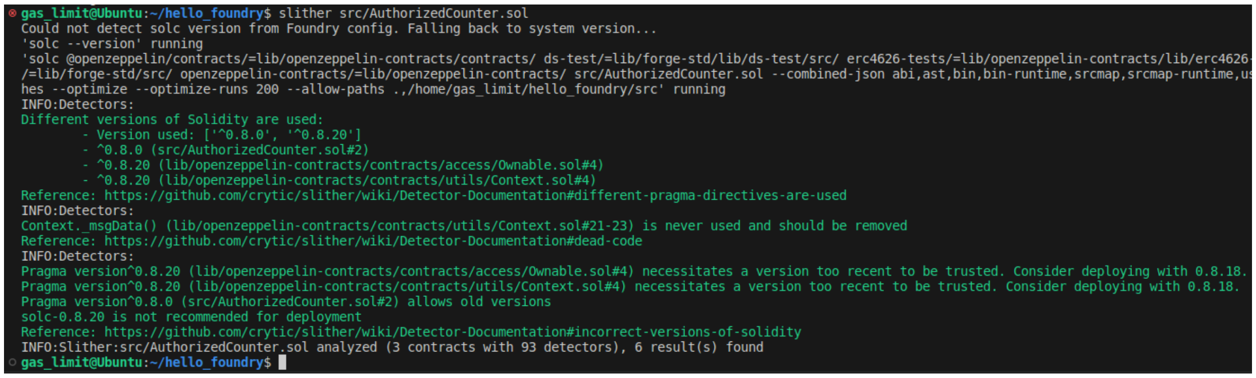
```
Switched global version to 0.8.4
⊗ gas_limit@Ubuntu:~/hello_foundry$ slither src/AuthorizedCounter.sol
  Could not detect solc version from Foundry config. Falling back to system version...
  'solc --version' running
  'solc @openzeppelin/contracts/=lib/openzeppelin-contracts/contracts/ ds-test/=lib/forge-std/lib/ds-test/src/ erc4626-te
  /=lib/forge-std/src/ openzeppelin-contracts/=lib/openzeppelin-contracts/ src/AuthorizedCounter.sol --combined-json abi,
  hes,compact-format --optimize --optimize-runs 200 --allow-paths .,/home/gas_limit/hello_foundry/src' running
  Compilation warnings/errors on src/AuthorizedCounter.sol:
  Error: Source file requires different compiler version (current compiler is 0.8.4+commit.c7e474f2.Linux.g++) - note tha
   released version
   --> lib/openzeppelin-contracts/contracts/access/Ownable.sol:4:1:
    |
  4 | pragma solidity ^0.8.20;
    | ^^^^^^^^^^^^^^^^^^^^^^^^


  Traceback (most recent call last):
    File "/home/gas_limit/.local/lib/python3.8/site-packages/crytic_compile/platform/solc.py", line 582, in _run_solc
      ret: Dict = json.loads(stdout)
    File "/usr/lib/python3.8/json/__init__.py", line 357, in loads
      return _default_decoder.decode(s)
    File "/usr/lib/python3.8/json/decoder.py", line 337, in decode
      obj, end = self.raw_decode(s, idx=_w(s, 0).end())
```

Openzeppelin’s Ownable library requires Solidity version 0.8.20



```
● gas_limit@Ubuntu:~/hello_foundry$ solc-select install 0.8.20
  Installing solc '0.8.20'...
  Version '0.8.20' installed.
● gas_limit@Ubuntu:~/hello_foundry$ solc-select use 0.8.20
  Switched global version to 0.8.20
○ gas_limit@Ubuntu:~/hello_foundry$ █
```

Switched compiler to 0.8.20 using ‘solc-select’

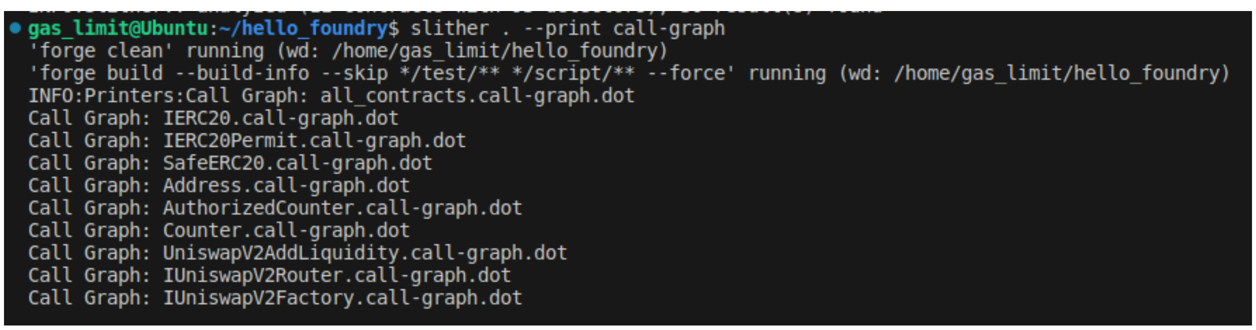


```
⊗ gas_limit@Ubuntu:~/hello_foundry$ slither src/AuthorizedCounter.sol
  Could not detect solc version from Foundry config. Falling back to system version...
  'solc --version' running
  'solc @openzeppelin/contracts/=lib/openzeppelin-contracts/contracts/ ds-test/=lib/forge-std/lib/ds-test/src/ erc4626-tests/=lib/openzeppelin-contracts/lib/erc4626-
  /=lib/forge-std/src/ openzeppelin-contracts/=lib/openzeppelin-contracts/ src/AuthorizedCounter.sol --combined-json abi,ast,bin,bin-runtime,srcmap,srcmap-runtime,us
  hes --optimize --optimize-runs 200 --allow-paths .,/home/gas_limit/hello_foundry/src' running
  INFO:Detectors:
  Different versions of Solidity are used:
          - Version used: ['^0.8.0', '^0.8.20']
          - ^0.8.0 (src/AuthorizedCounter.sol#2)
          - ^0.8.20 (lib/openzeppelin-contracts/contracts/access/Ownable.sol#4)
          - ^0.8.20 (lib/openzeppelin-contracts/contracts/utils/Context.sol#4)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
  INFO:Detectors:
  Context._msgData() (lib/openzeppelin-contracts/contracts/utils/Context.sol#21-23) is never used and should be removed
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
  INFO:Detectors:
  Pragma version^0.8.20 (lib/openzeppelin-contracts/contracts/access/Ownable.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.8.18.
  Pragma version^0.8.20 (lib/openzeppelin-contracts/contracts/utils/Context.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.8.18.
  Pragma version^0.8.0 (src/AuthorizedCounter.sol#2) allows old versions
  solc-0.8.20 is not recommended for deployment
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
  INFO:Slither:src/AuthorizedCounter.sol analyzed (3 contracts with 93 detectors), 6 result(s) found
○ gas_limit@Ubuntu:~/hello_foundry$ █
```

Analysis on individual file ran successfully

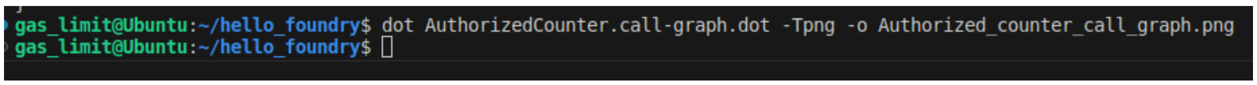
Creating call graph

Using '--print call-graph' flag



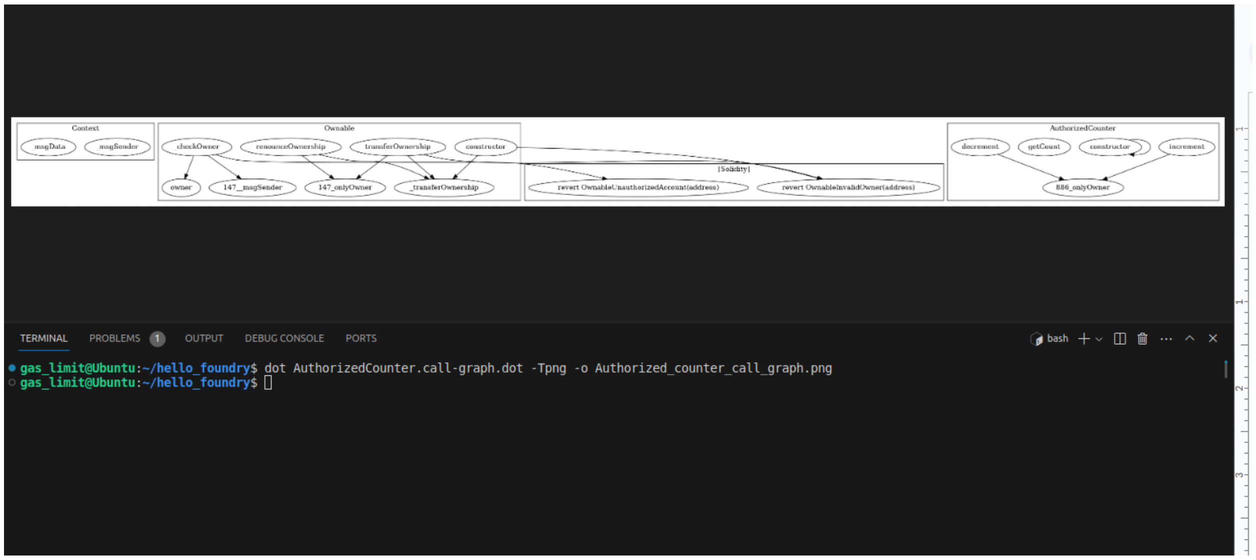
```
● gas_limit@Ubuntu:~/hello_foundry$ slither . --print call-graph
  'forge clean' running (wd: /home/gas_limit/hello_foundry)
  'forge build --build-info --skip */test/** */script/** --force' running (wd: /home/gas_limit/hello_foundry)
  INFO:Printers:Call Graph: all_contracts.call-graph.dot
  Call Graph: IERC20.call-graph.dot
  Call Graph: IERC20Permit.call-graph.dot
  Call Graph: SafeERC20.call-graph.dot
  Call Graph: Address.call-graph.dot
  Call Graph: AuthorizedCounter.call-graph.dot
  Call Graph: Counter.call-graph.dot
  Call Graph: UniswapV2AddLiquidity.call-graph.dot
  Call Graph: IUniswapV2Router.call-graph.dot
  Call Graph: IUniswapV2Factory.call-graph.dot
```

Using Graphviz to convert .dot into .svg



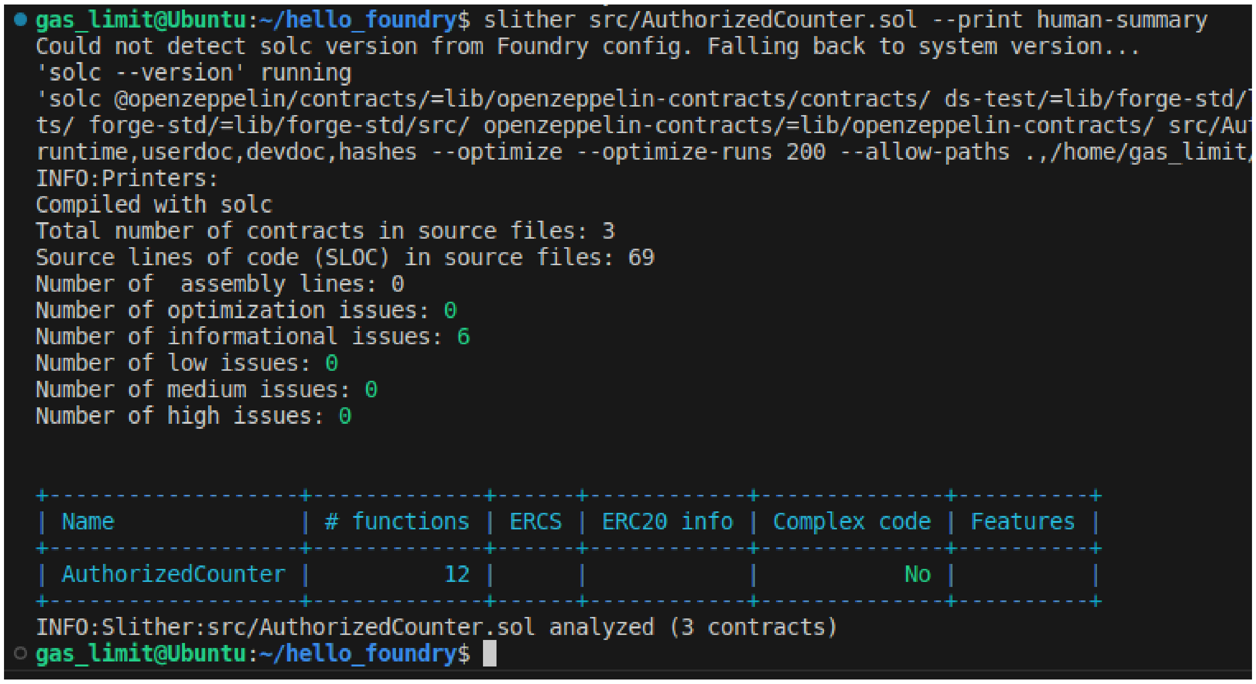
```
gas_limit@Ubuntu:~/hello_foundry$ dot AuthorizedCounter.call-graph.dot -Tpng -o Authorized_counter_call_graph.png
gas_limit@Ubuntu:~/hello_foundry$ 
```

Output call graph of AuthorizedCounter.sol



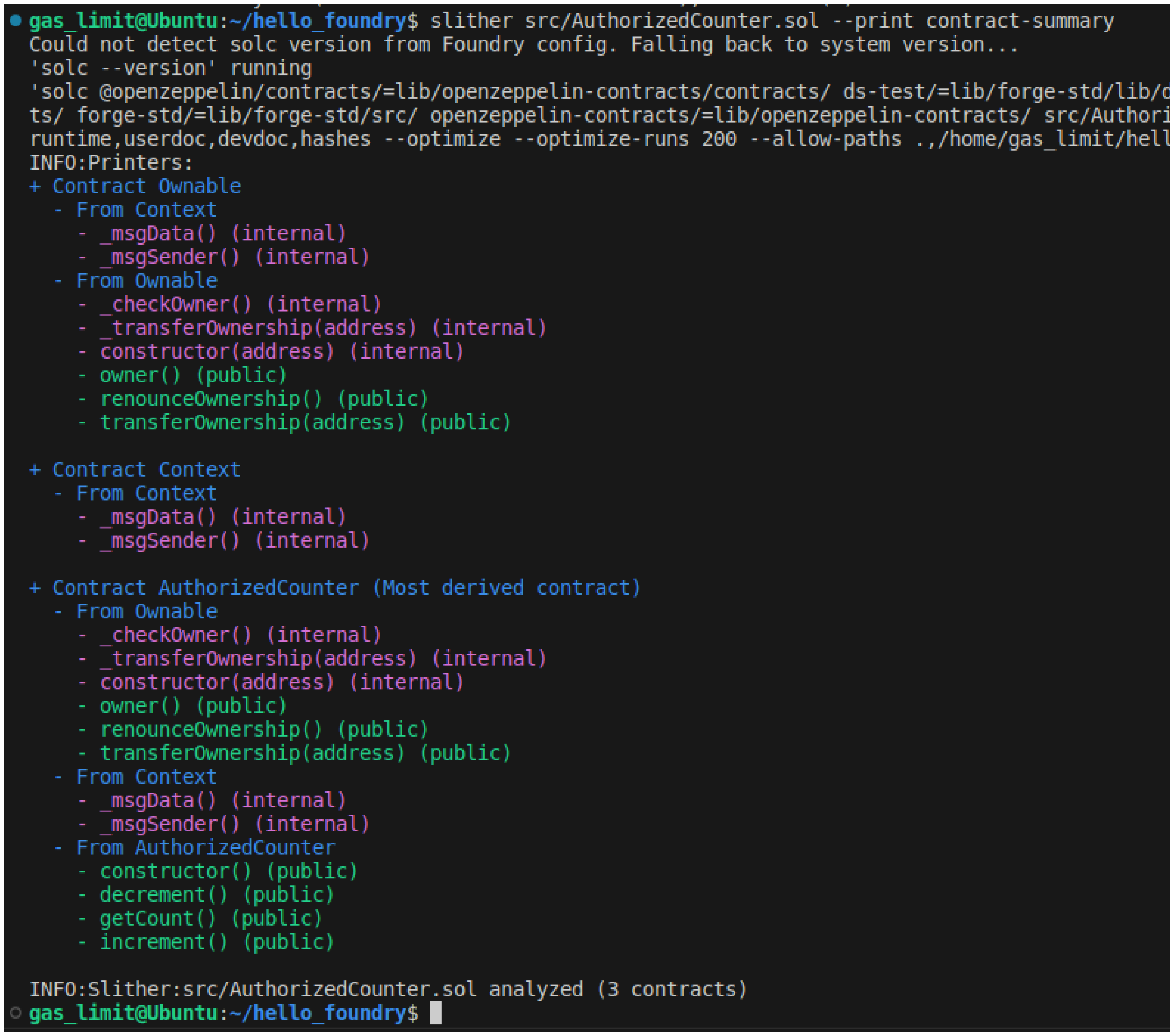
```
● gas_limit@Ubuntu:~/hello_foundry$ dot AuthorizedCounter.call-graph.dot -Tpng -o Authorized_counter_call_graph.png
○ gas_limit@Ubuntu:~/hello_foundry$ 
```

Printing a human readable summary of the contract scanned



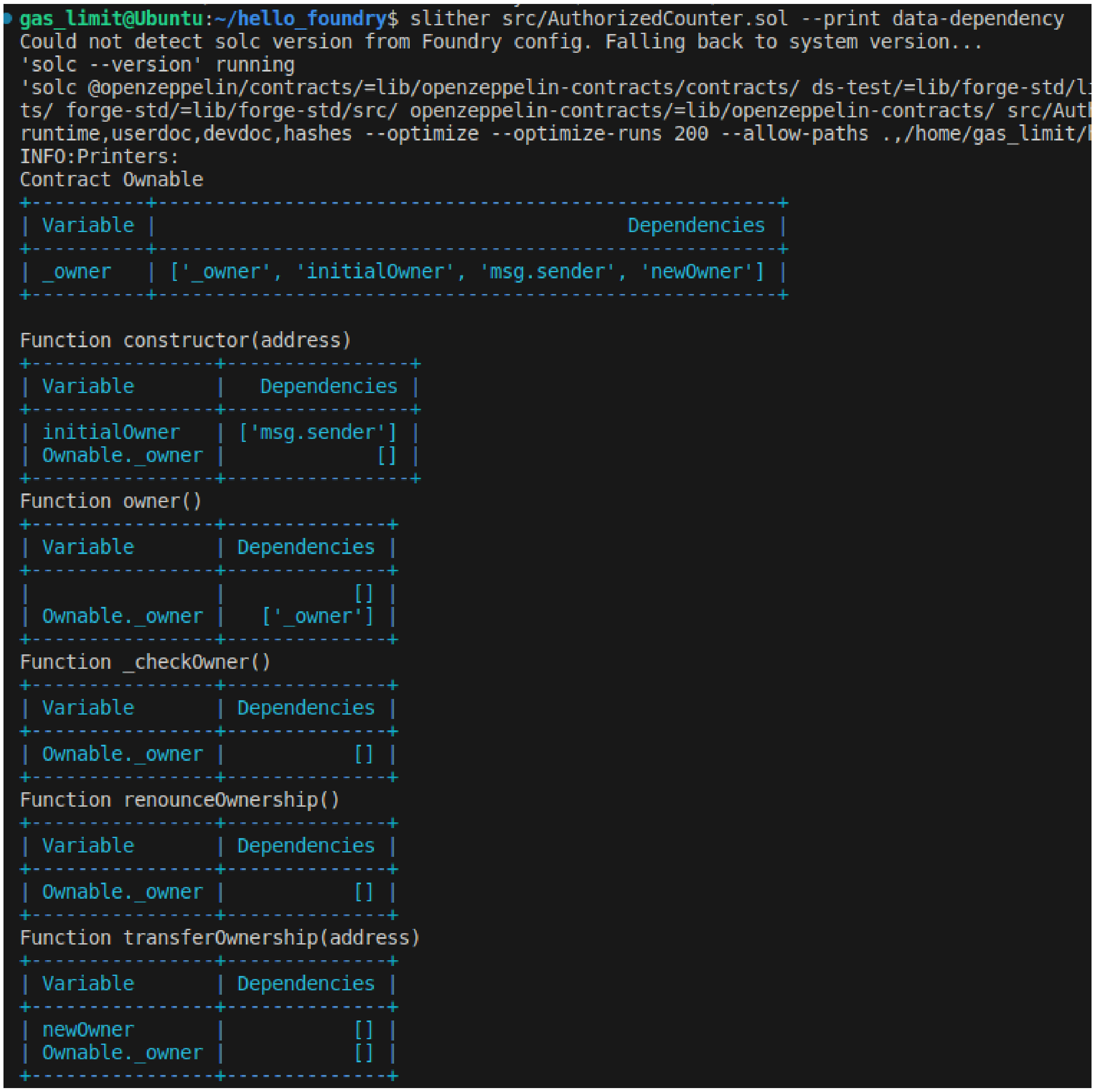
```
● gas_limit@Ubuntu:~/hello_foundry$ slither src/AuthorizedCounter.sol --print human-summary
  Could not detect solc version from Foundry config. Falling back to system version...
  'solc --version' running
  'solc @openzeppelin/contracts/=lib/openzeppelin-contracts/contracts/ ds-test/=lib/forge-std/l
  ts/ forge-std/=lib/forge-std/src/ openzeppelin-contracts/=lib/openzeppelin-contracts/ src/Au
  runtime,userdoc,devdoc,hashes --optimize --optimize-runs 200 --allow-paths .,/home/gas_limit,
  INFO:Printers:
  Compiled with solc
  Total number of contracts in source files: 3
  Source lines of code (SLOC) in source files: 69
  Number of  assembly lines: 0
  Number of optimization issues: 0
  Number of informational issues: 6
  Number of low issues: 0
  Number of medium issues: 0
  Number of high issues: 0


  +-------------------+-------------+------+------------+--------------+----------+
  | Name              | # functions | ERCS | ERC20 info | Complex code | Features |
  +-------------------+-------------+------+------------+--------------+----------+
  | AuthorizedCounter |          12 |      |            |           No |          |
  +-------------------+-------------+------+------------+--------------+----------+
  INFO:Slither:src/AuthorizedCounter.sol analyzed (3 contracts)
○ gas_limit@Ubuntu:~/hello_foundry$ █
```

Printing contract-summary using –print contract-summary flag

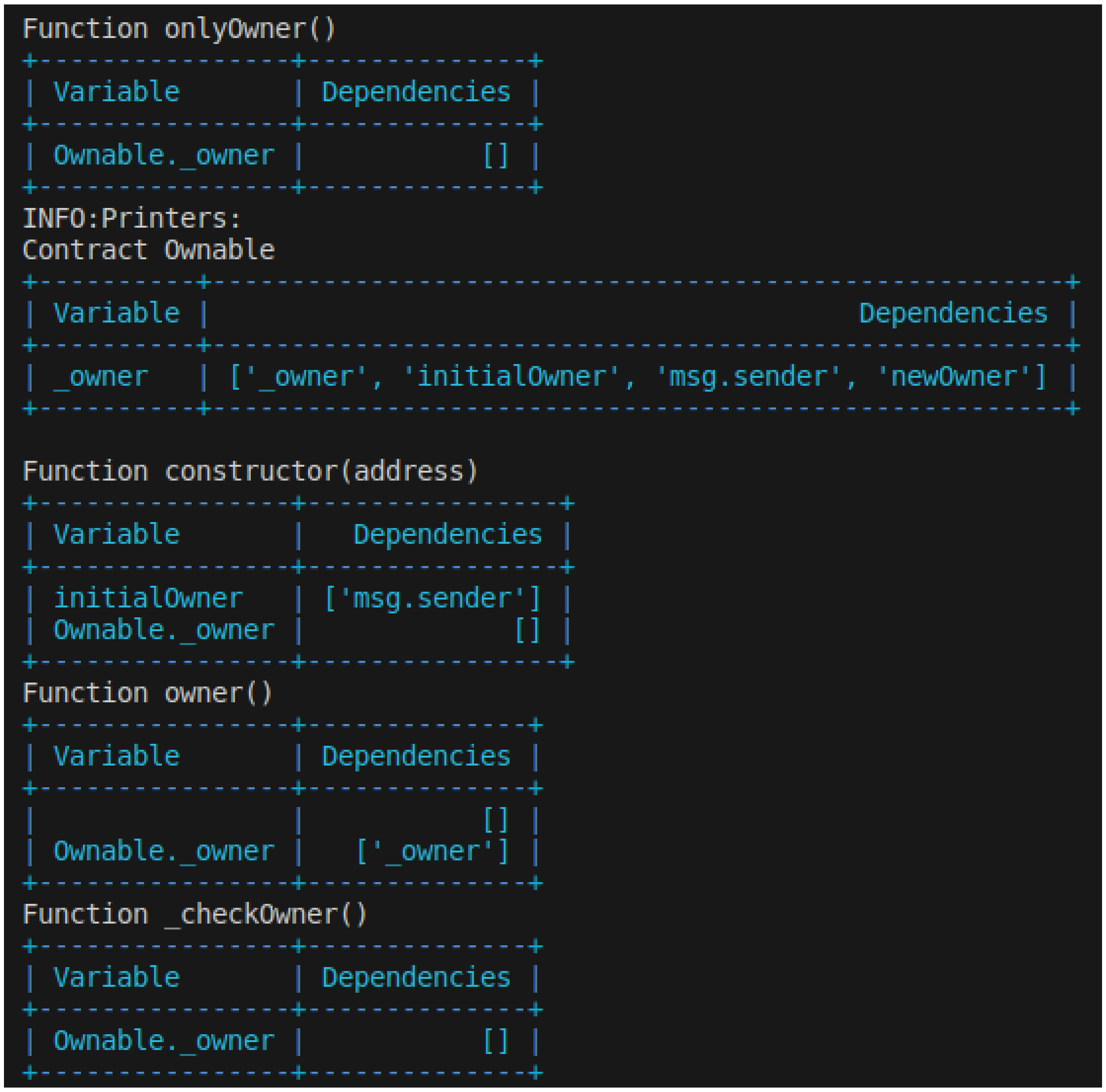


```
● gas_limit@Ubuntu:~/hello_foundry$ slither src/AuthorizedCounter.sol --print contract-summary
  Could not detect solc version from Foundry config. Falling back to system version...
  'solc --version' running
  'solc @openzeppelin/contracts/=lib/openzeppelin-contracts/contracts/ ds-test/=lib/forge-std/lib/d
  ts/ forge-std/=lib/forge-std/src/ openzeppelin-contracts/=lib/openzeppelin-contracts/ src/Authori
  runtime,userdoc,devdoc,hashes --optimize --optimize-runs 200 --allow-paths .,/home/gas_limit/hell
  INFO:Printers:
  + Contract Ownable
    - From Context
      - _msgData() (internal)
      - _msgSender() (internal)
    - From Ownable
      - _checkOwner() (internal)
      - _transferOwnership(address) (internal)
      - constructor(address) (internal)
      - owner() (public)
      - renounceOwnership() (public)
      - transferOwnership(address) (public)

  + Contract Context
    - From Context
      - _msgData() (internal)
      - _msgSender() (internal)

  + Contract AuthorizedCounter (Most derived contract)
    - From Ownable
      - _checkOwner() (internal)
      - _transferOwnership(address) (internal)
      - constructor(address) (internal)
      - owner() (public)
      - renounceOwnership() (public)
      - transferOwnership(address) (public)
    - From Context
      - _msgData() (internal)
      - _msgSender() (internal)
    - From AuthorizedCounter
      - constructor() (public)
      - decrement() (public)
      - getCount() (public)
      - increment() (public)

  INFO:Slither:src/AuthorizedCounter.sol analyzed (3 contracts)
○ gas_limit@Ubuntu:~/hello_foundry$ 
```

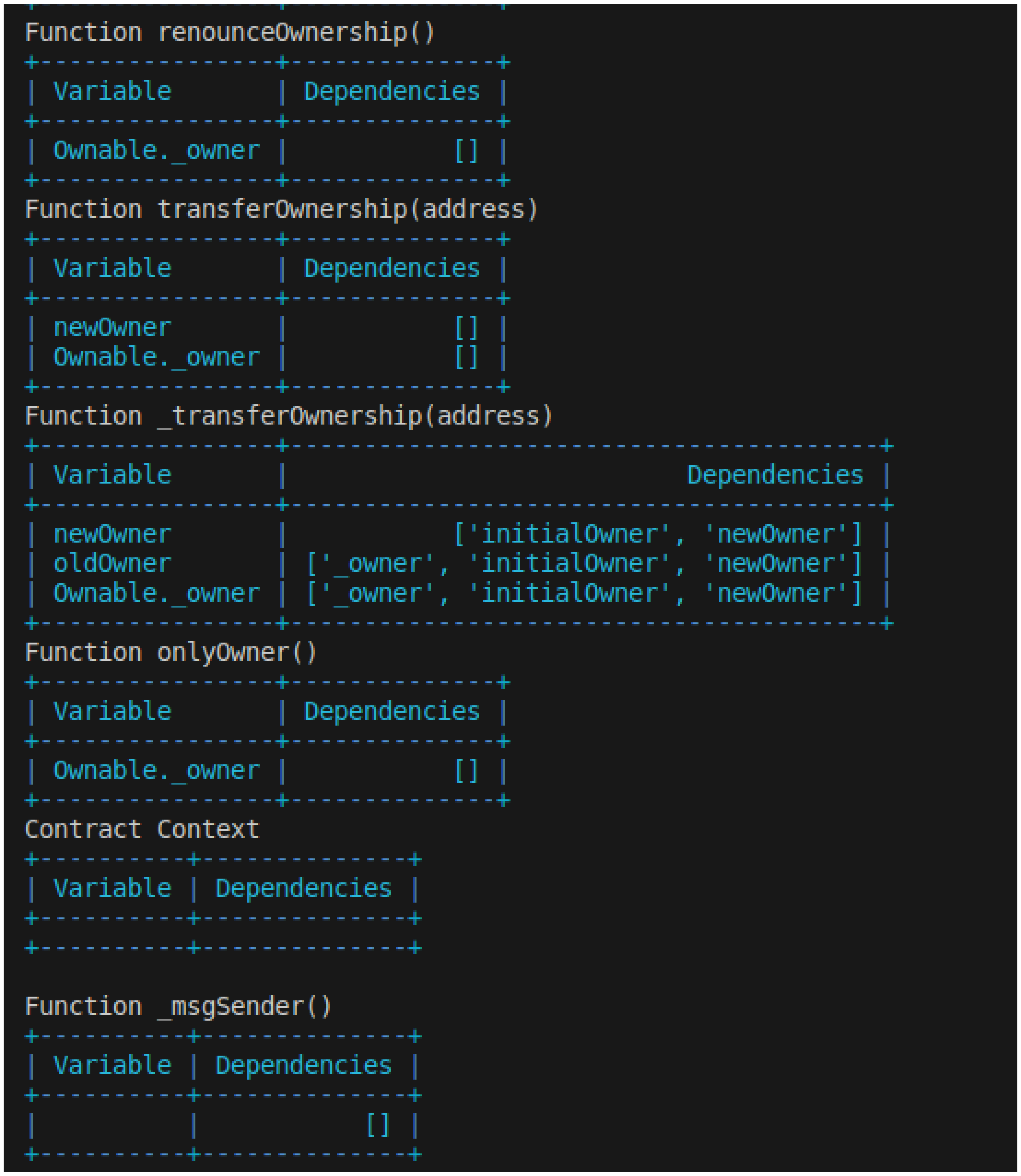
Printing data dependency using –print data-dependency flag



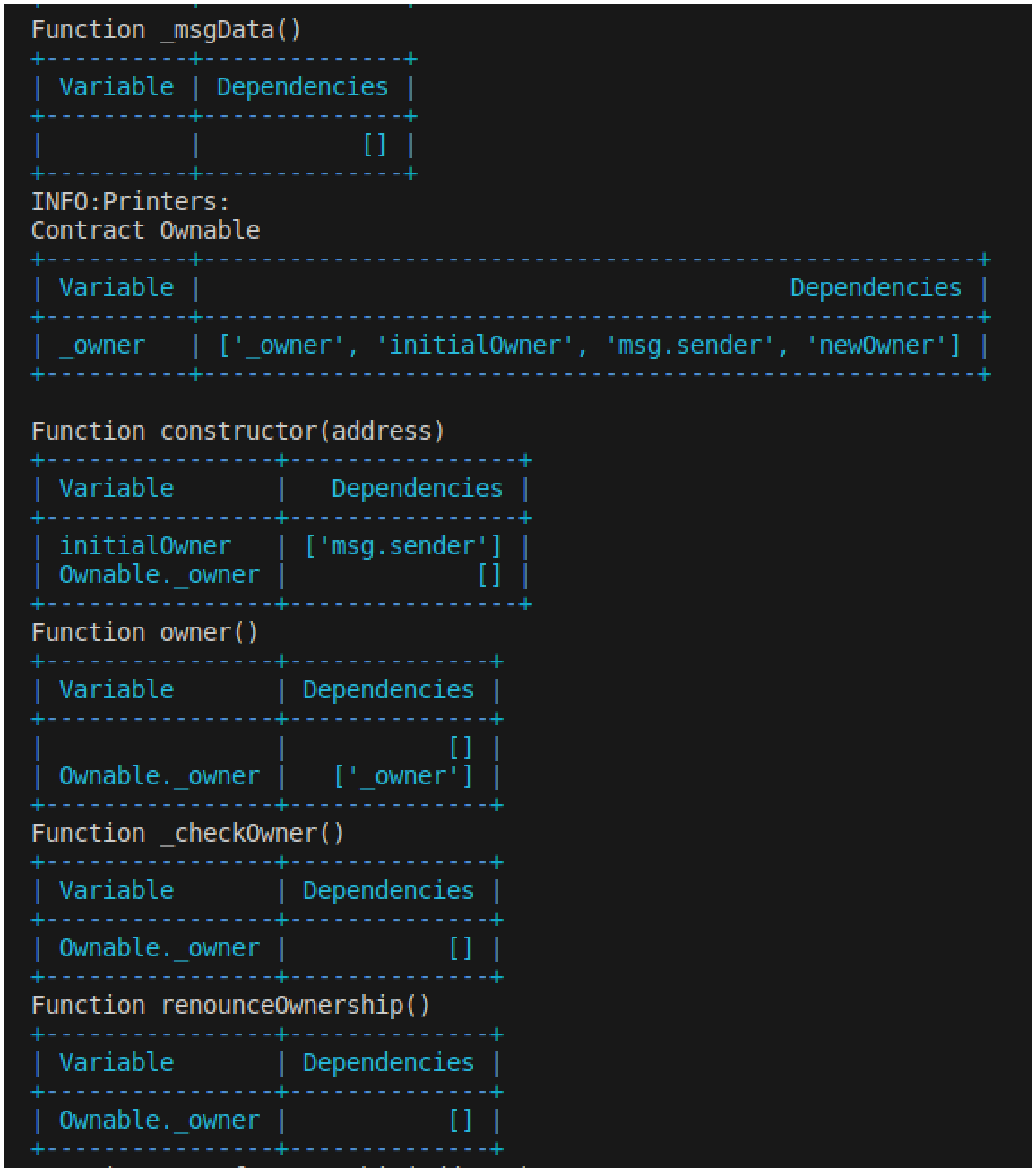
```
gas_limit@Ubuntu:~/hello_foundry$ slither src/AuthorizedCounter.sol --print data-dependency
Could not detect solc version from Foundry config. Falling back to system version...
'solc --version' running
'solc @openzeppelin/contracts/=lib/openzeppelin-contracts/contracts/ ds-test/=lib/forge-std/li
ts/ forge-std/=lib/forge-std/src/ openzeppelin-contracts/=lib/openzeppelin-contracts/ src/Auth
runtime,userdoc,devdoc,hashes --optimize --optimize-runs 200 --allow-paths .,/home/gas_limit/h
INFO:Printers:
Contract Ownable
+----------+-----------------------------------------------------+
| Variable |                                        Dependencies |
+----------+-----------------------------------------------------+
| _owner   | ['_owner', 'initialOwner', 'msg.sender', 'newOwner'] |
+----------+-----------------------------------------------------+

Function constructor(address)
+-----------------+----------------+
| Variable        |   Dependencies |
+-----------------+----------------+
| initialOwner    | ['msg.sender'] |
| Ownable._owner  |             [] |
+-----------------+----------------+
Function owner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
|                |           [] |
| Ownable._owner |   ['_owner'] |
+----------------+--------------+
Function _checkOwner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
Function renounceOwnership()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
Function transferOwnership(address)
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| newOwner       |           [] |
| Ownable._owner |           [] |
+----------------+--------------+
```



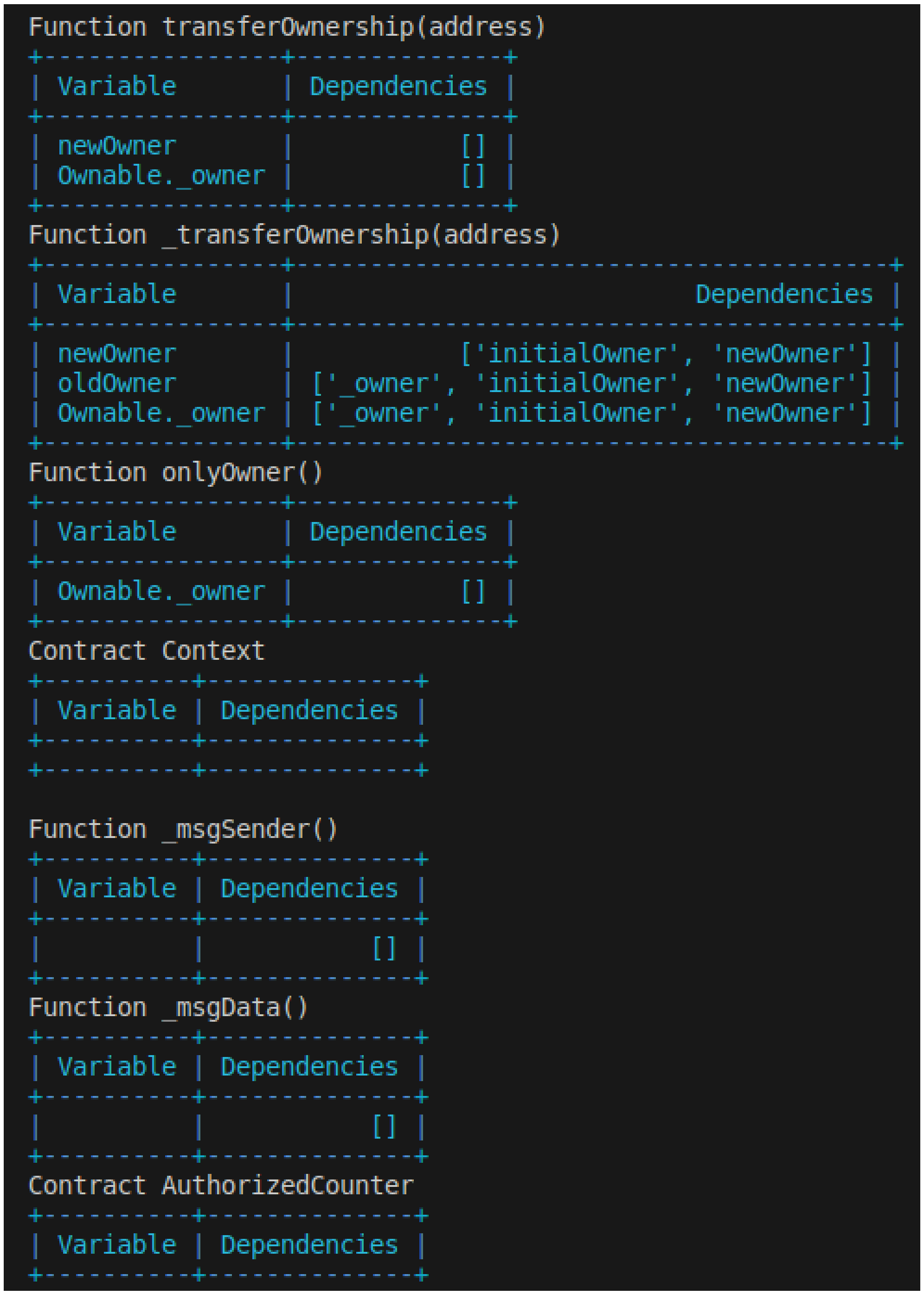
```
Function onlyOwner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
INFO:Printers:
Contract Ownable
+----------+----------------------------------------------------+
| Variable |                                       Dependencies |
+----------+----------------------------------------------------+
| _owner   | ['_owner', 'initialOwner', 'msg.sender', 'newOwner'] |
+----------+----------------------------------------------------+

Function constructor(address)
+----------------+----------------+
| Variable       |   Dependencies |
+----------------+----------------+
| initialOwner   | ['msg.sender'] |
| Ownable._owner |             [] |
+----------------+----------------+
Function owner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
|                |           [] |
| Ownable._owner |   ['_owner'] |
+----------------+--------------+
Function _checkOwner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
```



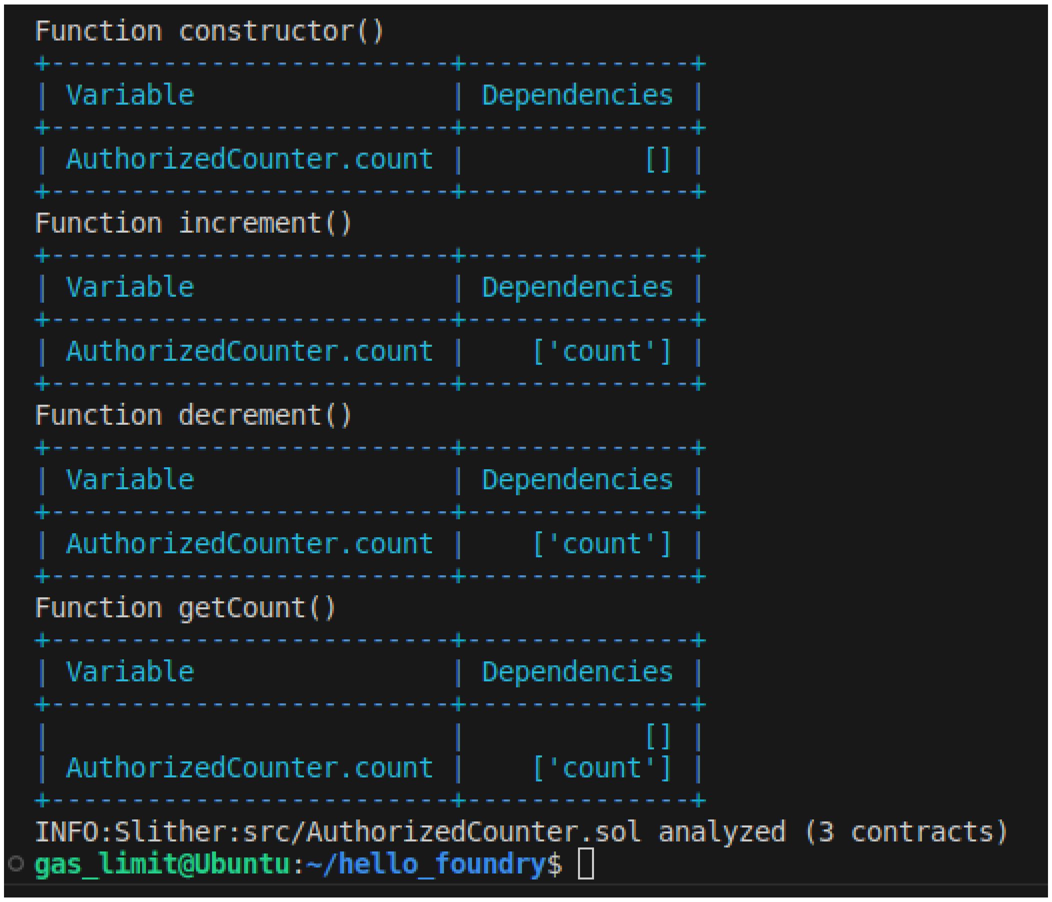
```
Function renounceOwnership()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
Function transferOwnership(address)
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| newOwner       |           [] |
| Ownable._owner |           [] |
+----------------+--------------+
Function _transferOwnership(address)
+----------------+--------------------------------------------+
| Variable       |                               Dependencies |
+----------------+--------------------------------------------+
| newOwner       |             ['initialOwner', 'newOwner'] |
| oldOwner       | ['_owner', 'initialOwner', 'newOwner'] |
| Ownable._owner | ['_owner', 'initialOwner', 'newOwner'] |
+----------------+--------------------------------------------+
Function onlyOwner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
Contract Context
+----------+--------------+
| Variable | Dependencies |
+----------+--------------+
+----------+--------------+

Function _msgSender()
+----------+--------------+
| Variable | Dependencies |
+----------+--------------+
|          |           [] |
+----------+--------------+
```



```
Function _msgData()
+----------+--------------+
| Variable | Dependencies |
+----------+--------------+
|          |           [] |
+----------+--------------+
INFO:Printers:
Contract Ownable
+----------+-----------------------------------------------------+
| Variable |                                        Dependencies |
+----------+-----------------------------------------------------+
| _owner   | ['_owner', 'initialOwner', 'msg.sender', 'newOwner'] |
+----------+-----------------------------------------------------+

Function constructor(address)
+----------------+----------------+
| Variable       |   Dependencies |
+----------------+----------------+
| initialOwner   | ['msg.sender'] |
| Ownable._owner |             [] |
+----------------+----------------+
Function owner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
|                |           [] |
| Ownable._owner |   ['_owner'] |
+----------------+--------------+
Function _checkOwner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
Function renounceOwnership()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
```

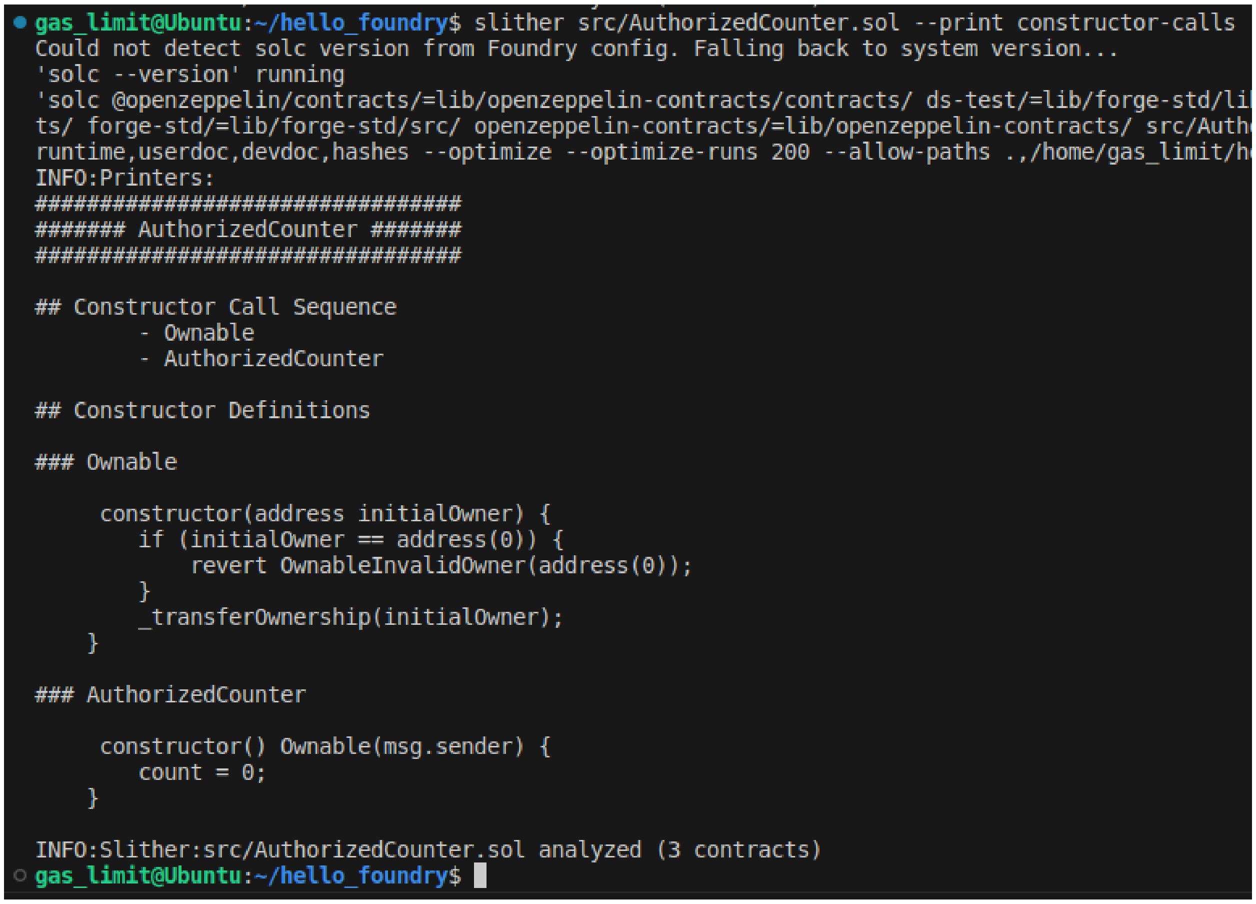


```
Function transferOwnership(address)
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| newOwner       |           [] |
| Ownable._owner |           [] |
+----------------+--------------+
Function _transferOwnership(address)
+----------------+--------------------------------------+
| Variable       |                         Dependencies |
+----------------+--------------------------------------+
| newOwner       |           ['initialOwner', 'newOwner'] |
| oldOwner       | ['_owner', 'initialOwner', 'newOwner'] |
| Ownable._owner | ['_owner', 'initialOwner', 'newOwner'] |
+----------------+--------------------------------------+
Function onlyOwner()
+----------------+--------------+
| Variable       | Dependencies |
+----------------+--------------+
| Ownable._owner |           [] |
+----------------+--------------+
Contract Context
+----------+--------------+
| Variable | Dependencies |
+----------+--------------+
+----------+--------------+

Function _msgSender()
+----------+--------------+
| Variable | Dependencies |
+----------+--------------+
|          |           [] |
+----------+--------------+
Function _msgData()
+----------+--------------+
| Variable | Dependencies |
+----------+--------------+
|          |           [] |
+----------+--------------+
Contract AuthorizedCounter
+----------+--------------+
| Variable | Dependencies |
+----------+--------------+
```



```
Function constructor()
+-------------------------+--------------+
| Variable                | Dependencies |
+-------------------------+--------------+
| AuthorizedCounter.count |           [] |
+-------------------------+--------------+
Function increment()
+-------------------------+--------------+
| Variable                | Dependencies |
+-------------------------+--------------+
| AuthorizedCounter.count |    ['count'] |
+-------------------------+--------------+
Function decrement()
+-------------------------+--------------+
| Variable                | Dependencies |
+-------------------------+--------------+
| AuthorizedCounter.count |    ['count'] |
+-------------------------+--------------+
Function getCount()
+-------------------------+--------------+
| Variable                | Dependencies |
+-------------------------+--------------+
|                         |           [] |
| AuthorizedCounter.count |    ['count'] |
+-------------------------+--------------+
INFO:Slither:src/AuthorizedCounter.sol analyzed (3 contracts)
gas_limit@Ubuntu:~/hello_foundry$
```

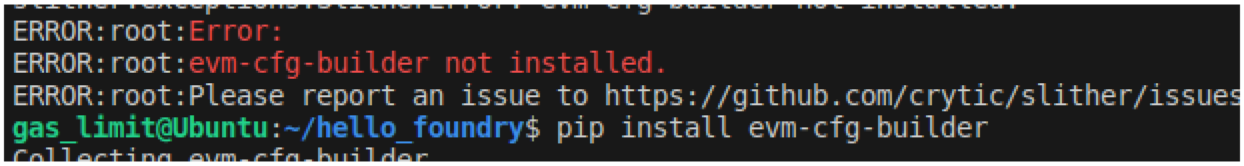
Printing constructor calls using –print constructor-calls flag



```
● gas_limit@Ubuntu:~/hello_foundry$ slither src/AuthorizedCounter.sol --print constructor-calls
  Could not detect solc version from Foundry config. Falling back to system version...
  'solc --version' running
  'solc @openzeppelin/contracts/=lib/openzeppelin-contracts/contracts/ ds-test/=lib/forge-std/li
  ts/ forge-std/=lib/forge-std/src/ openzeppelin-contracts/=lib/openzeppelin-contracts/ src/Auth
  runtime,userdoc,devdoc,hashes --optimize --optimize-runs 200 --allow-paths .,/home/gas_limit/h
  INFO:Printers:
  #################################
  ####### AuthorizedCounter #######
  #################################

  ## Constructor Call Sequence
          - Ownable
          - AuthorizedCounter

  ## Constructor Definitions

  ### Ownable

       constructor(address initialOwner) {
          if (initialOwner == address(0)) {
              revert OwnableInvalidOwner(address(0));
          }
          _transferOwnership(initialOwner);
      }

  ### AuthorizedCounter

       constructor() Ownable(msg.sender) {
          count = 0;
      }

  INFO:Slither:src/AuthorizedCounter.sol analyzed (3 contracts)
○ gas_limit@Ubuntu:~/hello_foundry$ 
```

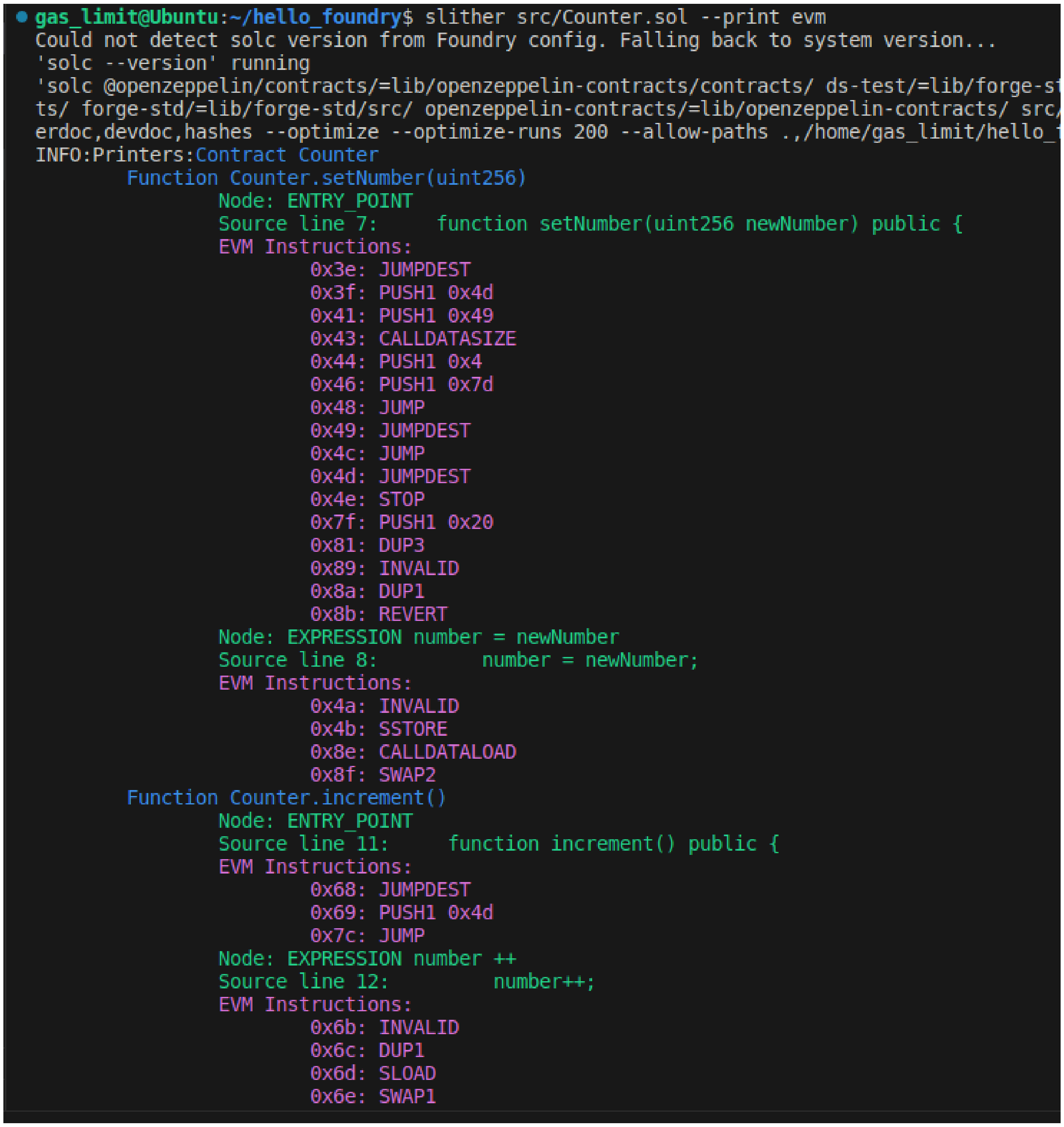
Printing the disassembly of a contract using –print evm tag

Had to install evm-cfg-builder



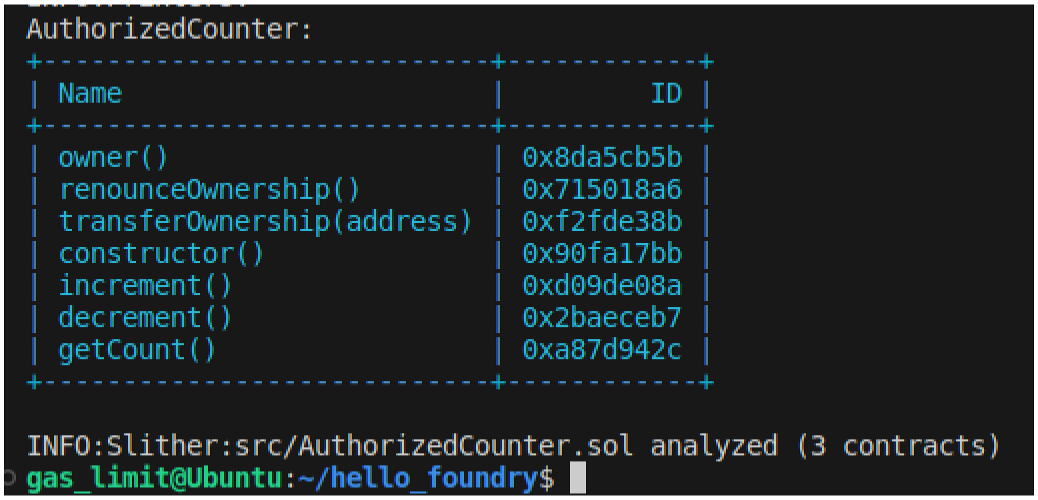
```
ERROR:root:Error:
ERROR:root:evm-cfg-builder not installed.
ERROR:root:Please report an issue to https://github.com/crytic/slither/issues
gas_limit@Ubuntu:~/hello_foundry$ pip install evm-cfg-builder
Collecting evm-cfg-builder
```

Executing disassembly analysis



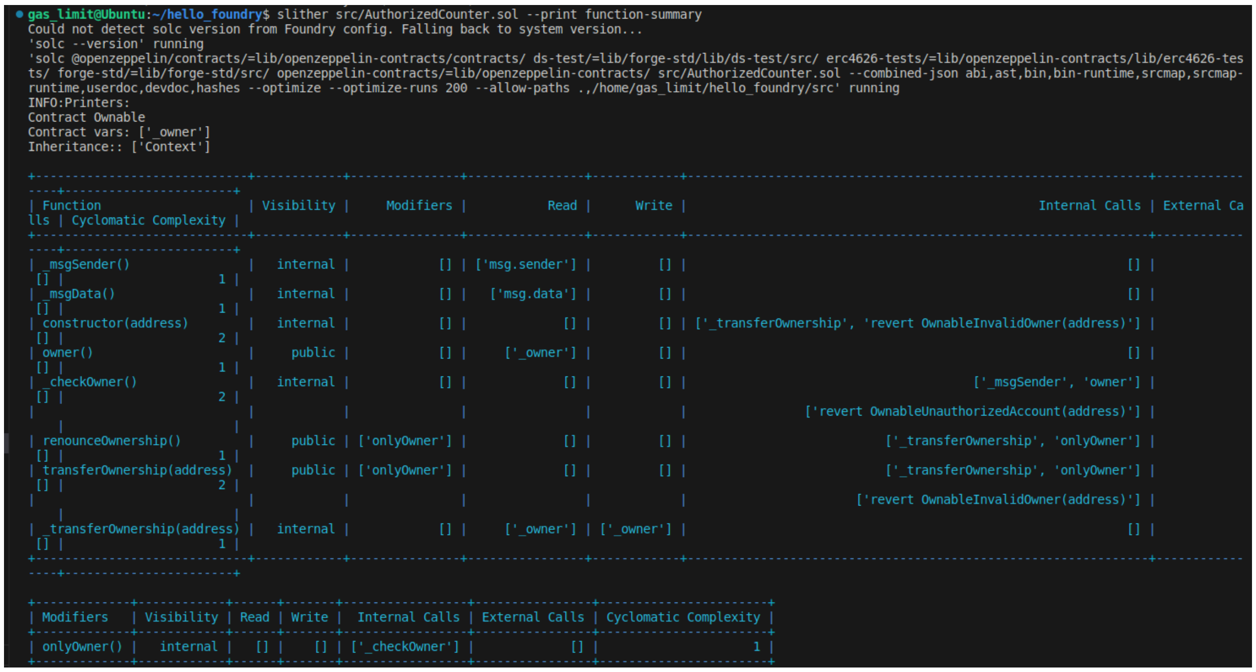
```
● gas_limit@Ubuntu:~/hello_foundry$ slither src/Counter.sol --print evm
  Could not detect solc version from Foundry config. Falling back to system version...
  'solc --version' running
  'solc @openzeppelin/contracts/=lib/openzeppelin-contracts/contracts/ ds-test/=lib/forge-st
  ts/ forge-std/=lib/forge-std/src/ openzeppelin-contracts/=lib/openzeppelin-contracts/ src/
  erdoc,devdoc,hashes --optimize --optimize-runs 200 --allow-paths .,/home/gas_limit/hello_f
  INFO:Printers:Contract Counter
          Function Counter.setNumber(uint256)
                  Node: ENTRY_POINT
                  Source line 7:     function setNumber(uint256 newNumber) public {
                  EVM Instructions:
                          0x3e: JUMPDEST
                          0x3f: PUSH1 0x4d
                          0x41: PUSH1 0x49
                          0x43: CALLDATASIZE
                          0x44: PUSH1 0x4
                          0x46: PUSH1 0x7d
                          0x48: JUMP
                          0x49: JUMPDEST
                          0x4c: JUMP
                          0x4d: JUMPDEST
                          0x4e: STOP
                          0x7f: PUSH1 0x20
                          0x81: DUP3
                          0x89: INVALID
                          0x8a: DUP1
                          0x8b: REVERT
                  Node: EXPRESSION number = newNumber
                  Source line 8:         number = newNumber;
                  EVM Instructions:
                          0x4a: INVALID
                          0x4b: SSTORE
                          0x8e: CALLDATALOAD
                          0x8f: SWAP2
          Function Counter.increment()
                  Node: ENTRY_POINT
                  Source line 11:     function increment() public {
                  EVM Instructions:
                          0x68: JUMPDEST
                          0x69: PUSH1 0x4d
                          0x7c: JUMP
                  Node: EXPRESSION number ++
                  Source line 12:         number++;
                  EVM Instructions:
                          0x6b: INVALID
                          0x6c: DUP1
                          0x6d: SLOAD
                          0x6e: SWAP1
```

Printing function signatures using –print function-id flag



```
AuthorizedCounter:
+-----------------------------+------------+
| Name                        |         ID |
+-----------------------------+------------+
| owner()                     | 0x8da5cb5b |
| renounceOwnership()         | 0x715018a6 |
| transferOwnership(address)  | 0xf2fde38b |
| constructor()               | 0x90fa17bb |
| increment()                 | 0xd09de08a |
| decrement()                 | 0x2baeceb7 |
| getCount()                  | 0xa87d942c |
+-----------------------------+------------+

INFO:Slither:src/AuthorizedCounter.sol analyzed (3 contracts)
gas_limit@Ubuntu:~/hello_foundry$
```

Printing function summaries using –print function-summary flag



```
gas_limit@Ubuntu:~/hello_foundry$ slither src/AuthorizedCounter.sol --print function-summary
Could not detect solc version from Foundry config. Falling back to system version...
'solc --version' running
'solc @openzeppelin/contracts/=lib/openzeppelin-contracts/contracts/ ds-test/=lib/forge-std/lib/ds-test/src/ erc4626-tests/=lib/openzeppelin-contracts/lib/erc4626-tes
ts/ forge-std/=lib/forge-std/src/ openzeppelin-contracts/=lib/openzeppelin-contracts/ src/AuthorizedCounter.sol --combined-json abi,ast,bin,bin-runtime,srcmap,srcmap-
runtime,userdoc,devdoc,hashes --optimize --optimize-runs 200 --allow-paths .,/home/gas_limit/hello_foundry/src' running
INFO:Printers:
Contract Ownable
Contract vars: ['_owner']
Inheritance:: ['Context']

+------------------------------+------------+---------------+---------------+------------+--------------------------------------------------------------+-----------
----+----------------------+
| Function                     | Visibility |     Modifiers |          Read |      Write |                                                Internal Calls | External Ca
lls | Cyclomatic Complexity |
+------------------------------+------------+---------------+---------------+------------+--------------------------------------------------------------+-----------
----+----------------------+
| _msgSender()                 |   internal |            [] | ['msg.sender'] |        [] |                                                            [] |
 [] |                    1 |
| _msgData()                   |   internal |            [] |   ['msg.data'] |        [] |                                                            [] |
 [] |                    1 |
| constructor(address)         |   internal |            [] |             [] |        [] | ['_transferOwnership', 'revert OwnableInvalidOwner(address)'] |
 [] |                    2 |
| owner()                      |     public |            [] |     ['_owner'] |        [] |                                                            [] |
 [] |                    1 |
| _checkOwner()                |   internal |            [] |             [] |        [] |                                    ['_msgSender', 'owner'] |
 [] |                    2 |
|                              |            |               |                |            |             ['revert OwnableUnauthorizedAccount(address)'] |
    |                      |
| renounceOwnership()          |     public | ['onlyOwner'] |             [] |        [] |                          ['_transferOwnership', 'onlyOwner'] |
 [] |                    1 |
| transferOwnership(address)   |     public | ['onlyOwner'] |             [] |        [] |                          ['_transferOwnership', 'onlyOwner'] |
 [] |                    2 |
|                              |            |               |                |            |                   ['revert OwnableInvalidOwner(address)'] |
    |                      |
| _transferOwnership(address)  |   internal |            [] |     ['_owner'] | ['_owner'] |                                                            [] |
 [] |                    1 |
+------------------------------+------------+---------------+---------------+------------+--------------------------------------------------------------+-----------
----+----------------------+

+-------------+------------+------+-------+------------------+----------------+-----------------------+
| Modifiers   | Visibility | Read | Write |   Internal Calls | External Calls | Cyclomatic Complexity |
+-------------+------------+------+-------+------------------+----------------+-----------------------+
| onlyOwner() |   internal |   [] |    [] | ['_checkOwner'] |             [] |                     1 |
+-------------+------------+------+-------+------------------+----------------+-----------------------+
```

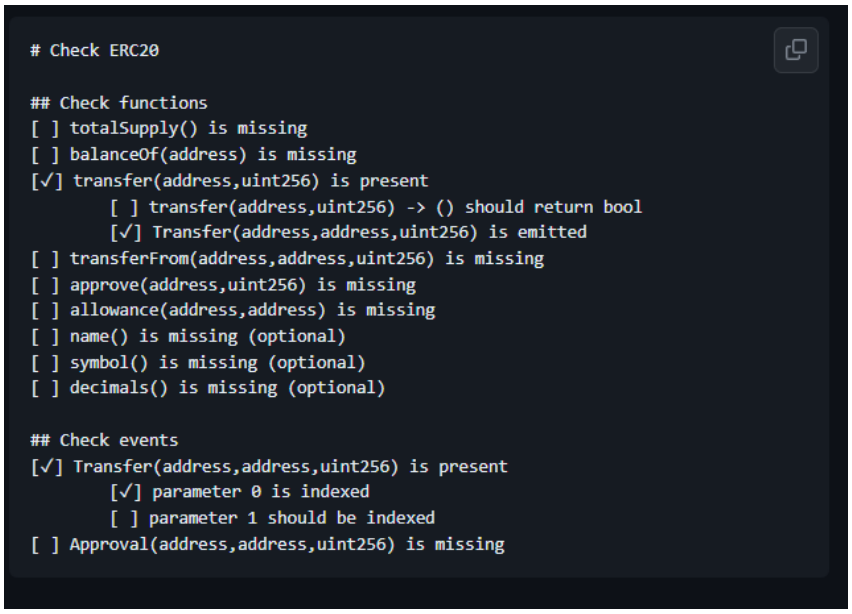
Abstraction for Brevity

There are many more printer tools available, below is a full list:

https://github.com/crytic/slither/wiki/Printer-documentation

Some contract tools I found to be very useful:

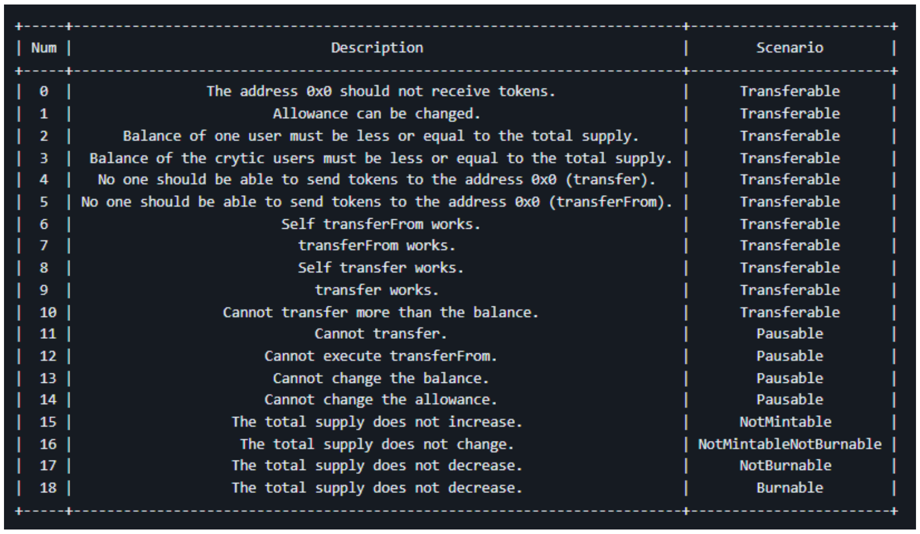
Check the ERC's conformance
https://github.com/crytic/slither/wiki/ERC-Conformance



```
# Check ERC20

## Check functions
[ ] totalSupply() is missing
[ ] balanceOf(address) is missing
[✓] transfer(address,uint256) is present
        [ ] transfer(address,uint256) -> () should return bool
        [✓] Transfer(address,address,uint256) is emitted
[ ] transferFrom(address,address,uint256) is missing
[ ] approve(address,uint256) is missing
[ ] allowance(address,address) is missing
[ ] name() is missing (optional)
[ ] symbol() is missing (optional)
[ ] decimals() is missing (optional)

## Check events
[✓] Transfer(address,address,uint256) is present
        [✓] parameter 0 is indexed
        [ ] parameter 1 should be indexed
[ ] Approval(address,address,uint256) is missing
```

Automated Unit tests

https://github.com/crytic/slither/wiki/Property-generation



```
+-----+-----------------------------------------------------------------------+-----------------------+
| Num |                              Description                              |        Scenario       |
+-----+-----------------------------------------------------------------------+-----------------------+
|  0  |                The address 0x0 should not receive tokens.             |      Transferable     |
|  1  |                       Allowance can be changed.                       |      Transferable     |
|  2  |      Balance of one user must be less or equal to the total supply.   |      Transferable     |
|  3  |  Balance of the crytic users must be less or equal to the total supply. |    Transferable     |
|  4  |   No one should be able to send tokens to the address 0x0 (transfer).  |     Transferable     |
|  5  | No one should be able to send tokens to the address 0x0 (transferFrom). |    Transferable     |
|  6  |                        Self transferFrom works.                       |      Transferable     |
|  7  |                          transferFrom works.                          |      Transferable     |
|  8  |                          Self transfer works.                         |      Transferable     |
|  9  |                            transfer works.                            |      Transferable     |
|  10 |                   Cannot transfer more than the balance.              |      Transferable     |
|  11 |                            Cannot transfer.                           |        Pausable       |
|  12 |                      Cannot execute transferFrom.                     |        Pausable       |
|  13 |                        Cannot change the balance.                     |        Pausable       |
|  14 |                       Cannot change the allowance.                    |        Pausable       |
|  15 |                    The total supply does not increase.                |      NotMintable      |
|  16 |                     The total supply does not change.                 | NotMintableNotBurnable |
|  17 |                    The total supply does not decrease.                |      NotBurnable      |
|  18 |                    The total supply does not decrease.                |        Burnable       |
+-----+-----------------------------------------------------------------------+-----------------------+
```

Codebase flattening (putting all files into one)

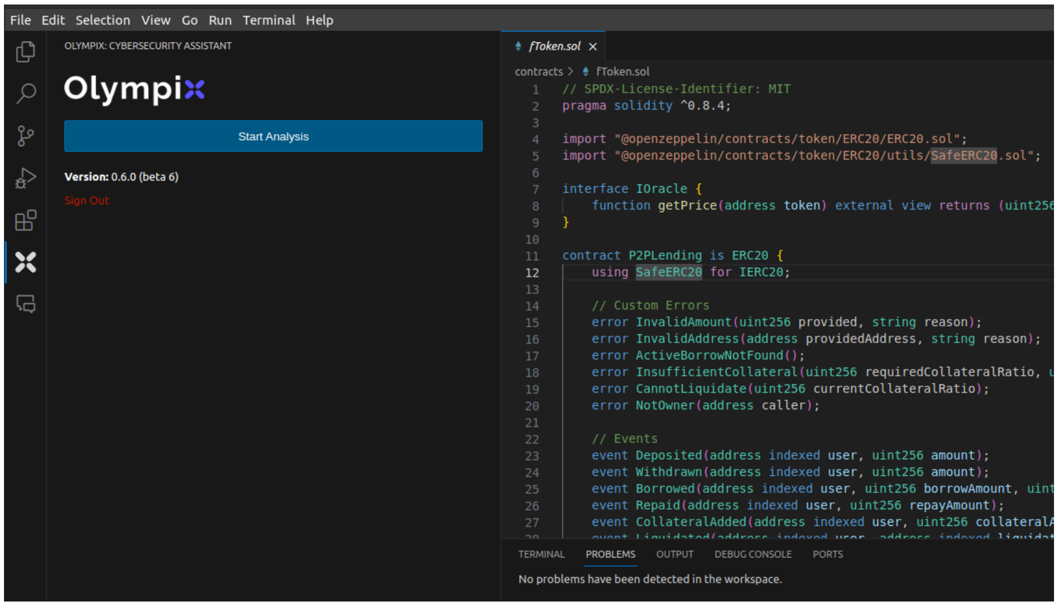
https://github.com/crytic/slither/wiki/Contract-Flattening

Olympix

A Cybersecurity Assistant for Web 3 projects written in Solidity. This extension allows for integration into Visual Studio Code for Olympix. The Olympix extension performs static code analysis on projects written in Solidity. By using the Olympix extension in Visual Studio Code, developers can find potentially dangerous vulnerabilities while developing in real-time.

For my analysis, I selected a contract I developed for the ETHGlobal Online 2023 hackathon. This contract is a lending pool designed for deployment on Filecoin FVM and draws inspiration from the AAVE and Compound lending models. Its core functionality allows users to deposit stablecoins, earning lending fees in return. Concurrently, borrowers can secure loans using Filecoin as collateral, paying a variable fee based on the pool's utilization rate. If borrowers' collateral falls below the required threshold, they face potential liquidation. To facilitate this, liquidators are incentivized with rewards for successful liquidations. To avoid liquidation, borrowers can supplement their collateral. Partial liquidations are also an option until borrowers maintain the requisite collateral level. Integral to the contract's operations is a singular price oracle native to the Filecoin FVM network.
Link to contract: https://github.com/JustUzair/ethonline-filecoin-project/blob/master/backend
For result comparison, slither tests can be seen here.

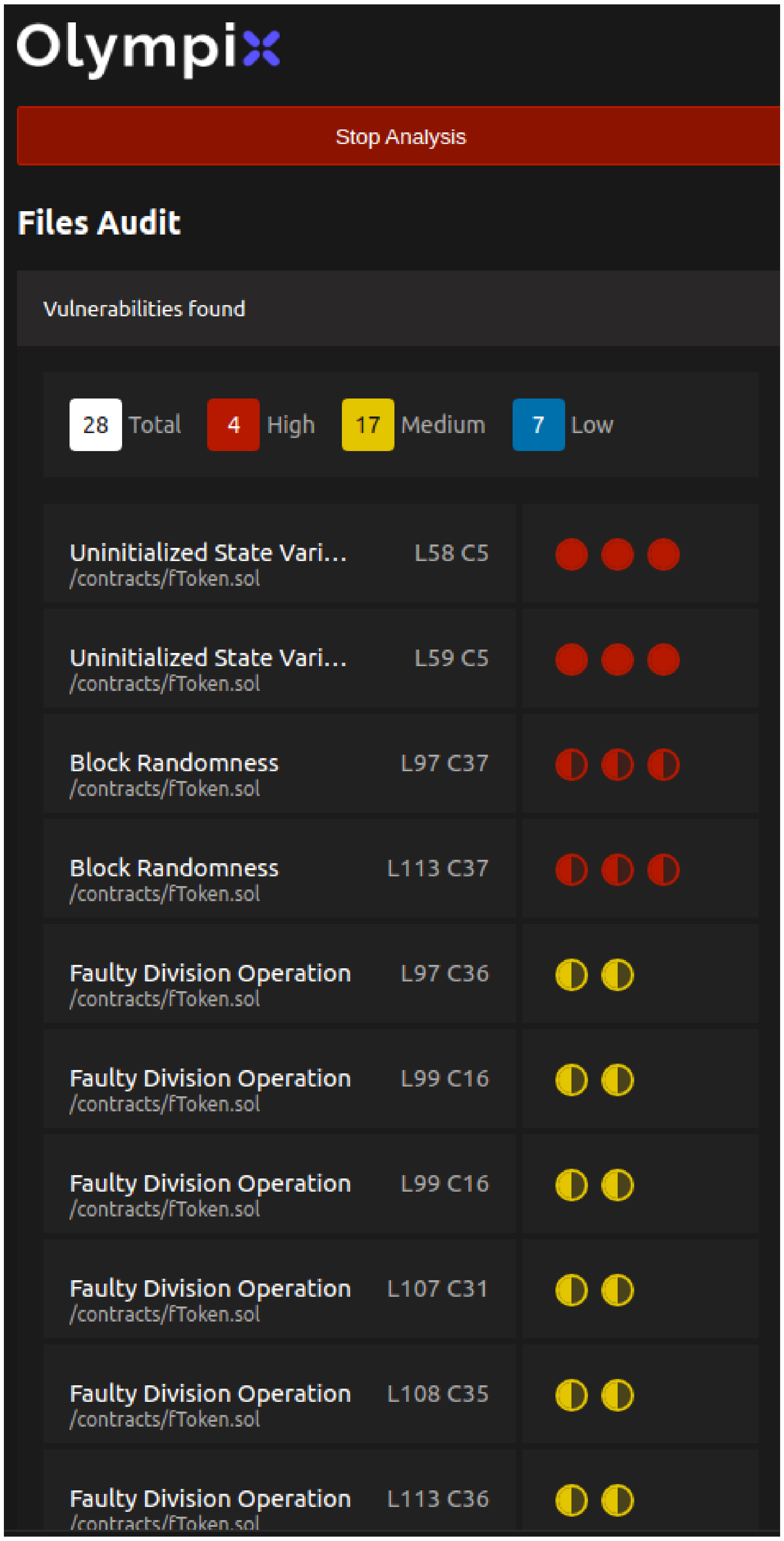
Below is the extension opened on the left before analysis


File Edit Selection View Go Run Terminal Help
OLYMPIX: CYBERSECURITY ASSISTANT
Olympix
Start Analysis
Version: 0.6.0 (beta 6)
Sign Out
fToken.sol
contracts > fToken.sol
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.4;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol";

interface IOracle {
    function getPrice(address token) external view returns (uint256
}

contract P2PLending is ERC20 {
    using SafeERC20 for IERC20;

    // Custom Errors
    error InvalidAmount(uint256 provided, string reason);
    error InvalidAddress(address providedAddress, string reason);
    error ActiveBorrowNotFound();
    error InsufficientCollateral(uint256 requiredCollateralRatio, u
    error CannotLiquidate(uint256 currentCollateralRatio);
    error NotOwner(address caller);

    // Events
    event Deposited(address indexed user, uint256 amount);
    event Withdrawn(address indexed user, uint256 amount);
    event Borrowed(address indexed user, uint256 borrowAmount, uint
    event Repaid(address indexed user, uint256 repayAmount);
    event CollateralAdded(address indexed user, uint256 collateralA
```

No problems have been detected in the workspace.

I ran the test and the extension propagated some findings shown below and in the terminal


Olympix

Stop Analysis

Files Audit

Vulnerabilities found

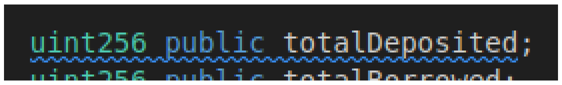
28 Total 4 High 17 Medium 7 Low

Uninitialized State Vari… /contracts/fToken.sol	L58 C5	● ● ●
Uninitialized State Vari… /contracts/fToken.sol	L59 C5	● ● ●
Block Randomness /contracts/fToken.sol	L97 C37	◐ ◐ ◐
Block Randomness /contracts/fToken.sol	L113 C37	◐ ◐ ◐
Faulty Division Operation /contracts/fToken.sol	L97 C36	◐ ◐
Faulty Division Operation /contracts/fToken.sol	L99 C16	◐ ◐
Faulty Division Operation /contracts/fToken.sol	L99 C16	◐ ◐
Faulty Division Operation /contracts/fToken.sol	L107 C31	◐ ◐
Faulty Division Operation /contracts/fToken.sol	L108 C35	◐ ◐
Faulty Division Operation /contracts/fToken.sol	L113 C36	◐ ◐

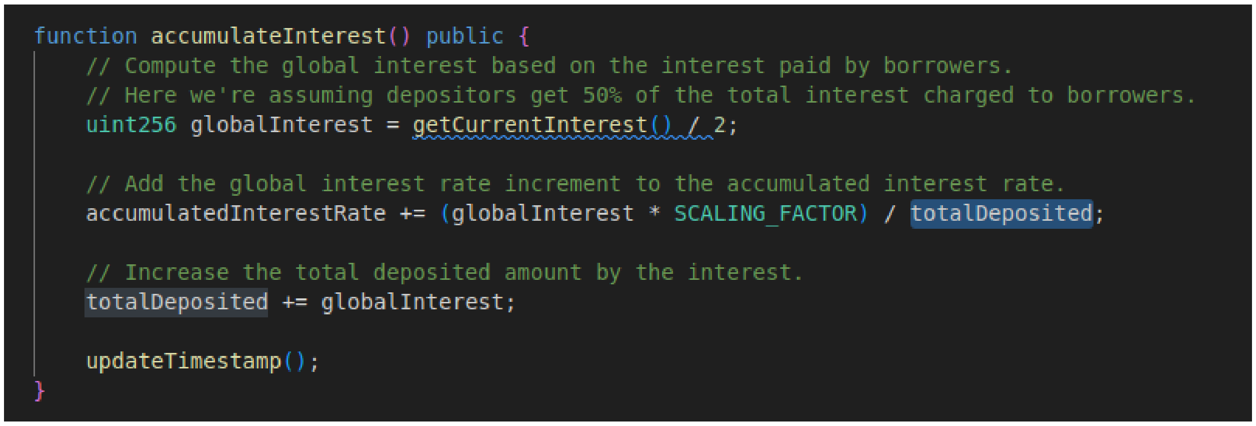
The findings:

High: Using uninitialized state variables may lead to unexpected behavior. (2 instances)

Was flagged because I did not initialize a value, this is a valid find and I understand why it can be a problem. For example, some of my functions would fail because I'd be dividing by zero if no liquidity was deposited.



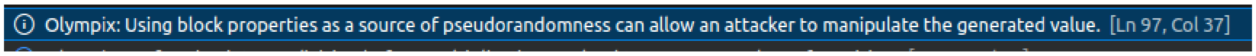
```
uint256 public totalDeposited;
```



```
function accumulateInterest() public {
    // Compute the global interest based on the interest paid by borrowers.
    // Here we're assuming depositors get 50% of the total interest charged to borrowers.
    uint256 globalInterest = getCurrentInterest() / 2;

    // Add the global interest rate increment to the accumulated interest rate.
    accumulatedInterestRate += (globalInterest * SCALING_FACTOR) / totalDeposited;

    // Increase the total deposited amount by the interest.
    totalDeposited += globalInterest;

    updateTimestamp();
}
```

A possible fix to the contract would be to limit the accumulateInterest() function to be called only when there is >0 liquidity.

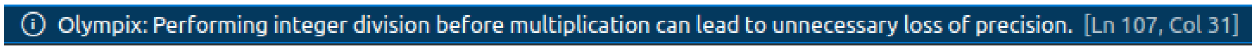
High: Block Randomness (2 instances)



```
Olympix: Using block properties as a source of pseudorandomness can allow an attacker to manipulate the generated value. [Ln 97, Col 37]
```

Block randomness can indeed be slightly manipulated by miners, and it made me rethink how certain factors of each event may be affected. For example, in my lending protocol, it adjusts interest rates based on supply and demand dynamics, and with specific timeframes. Manipulating block.timestamp could, in theory, impact these adjustments, though it would generally be a minor influence.

Medium: Faulty Division Operation (17 instances)

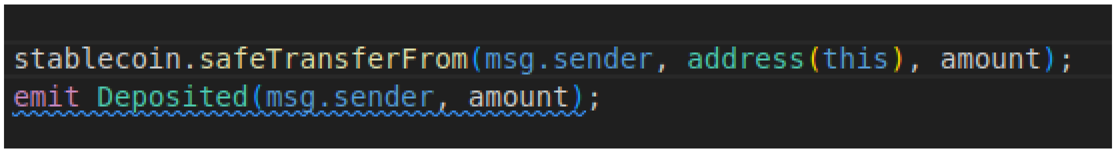


```
Olympix: Performing integer division before multiplication can lead to unnecessary loss of precision. [Ln 107, Col 31]
```

At first glance, I believed the alert to be a false positive. While dividing before multiplying can indeed result in precision loss, the frequent flags every time I multiplied seemed excessive. However, a deeper examination of the code revealed the warnings to be justified. In my contract, I initiate division early within the functions—a practice not recommended in Solidity. To minimize precision loss, division should be postponed as long as feasible. I've contacted the developers of the static analysis tool to discuss renaming the alert or clarifying its criteria and context.

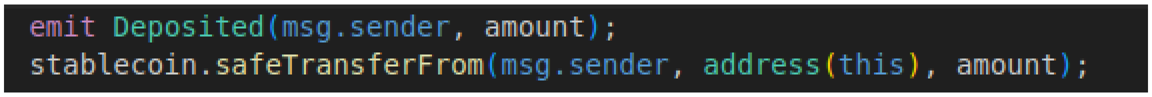
Low: Reentrant functions which emit events after making an external call may lead to out-of-order events (7 instances)

This was triggered most likely because I was emitting an event after making an external call to a token contract. While the potential for exploitation is very low, it does not follow best solidity practices. After further research, I learned that any off-chain system that's listening to these events might interpret certain events in an incorrect sequence, which can lead to unexpected behaviors or logical inconsistencies in the external systems monitoring the events.
In solidity, following the checks-effects-interactions pattern is the best way to prevent reentrancy. While there is not a reentrancy risk, I've learned events are considered part of the effects. I can fix this finding by emitting the event before making the external call.
From this:



```
stablecoin.safeTransferFrom(msg.sender, address(this), amount);
emit Deposited(msg.sender, amount);
```

to this which does not trigger the finding:



```
emit Deposited(msg.sender, amount);
stablecoin.safeTransferFrom(msg.sender, address(this), amount);
```

Pyrometer

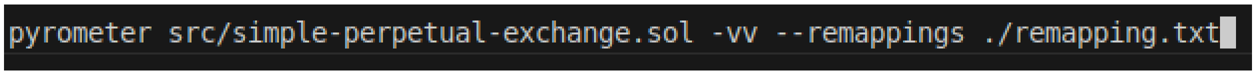
Description:
Pyrometer is an emerging security tool designed to enhance the security of smart contracts written in Solidity, the programming language primarily used for Ethereum blockchain development. It's currently in the BETA phase and is optimized for contracts crafted with Solidity version 0.8.x, although it may not fully support all language constructs and exceptional cases. Pyrometer utilizes a blend of symbolic execution, abstract interpretation, and static analysis techniques to analyze and flag potential security vulnerabilities. While it currently focuses on Solidity, the tool is constructed with a foundation that could enable future support for other languages targeting the Ethereum Virtual Machine (EVM). Pyrometer is being developed with practical application in mind, seeking to aid both developers in writing secure code and auditors in reviewing it, while consciously steering clear of becoming entangled in purely academic debates.

Subject:
The subject of this static analysis is the Solidity smart contract I wrote last week designed to function as a decentralized perpetual exchange, as per the specifications of Owen Thurm's Security Course's first mission. Complying with Solidity ^0.8.4, the contract incorporates the SafeERC20 library from OpenZeppelin for secure token transactions and the AggregatorV3Interface from Chainlink to fetch real-time BTC prices. Core functionalities enable liquidity providers to deposit and withdraw liquidity, while traders can open, increase the size, and add collateral to their perpetual positions in BTC. Notably, the smart contract enforces restrictions that prevent traders from using more than an allowed percentage of the total liquidity and prohibits the withdrawal of liquidity earmarked for open positions. The hardcoded USDC price within the contract simplifies the collateralization calculations, though it may not reflect true market conditions.
The contract refrains from implementing transaction fees and lacks mechanisms for decreasing, closing, or liquidating positions, which are identified as areas for future development. Other potential enhancements could include dynamic pricing for USDC to reflect live market values, incorporation of fee structures for platform sustainability, and advanced risk management features such as automated liquidation to safeguard the system's solvency. The current state of the contract establishes a fundamental architecture for a decentralized trading platform, with essential safeguards and parameters that align with best practices in smart contract design for financial operations. It prioritizes security and simplicity while laying down a solid groundwork for introducing more sophisticated functionalities in line with the evolving demands of decentralized finance infrastructure.

Link to the repository:
https://github.com/gas-limit/PURPetual-exchange.git

Test Results

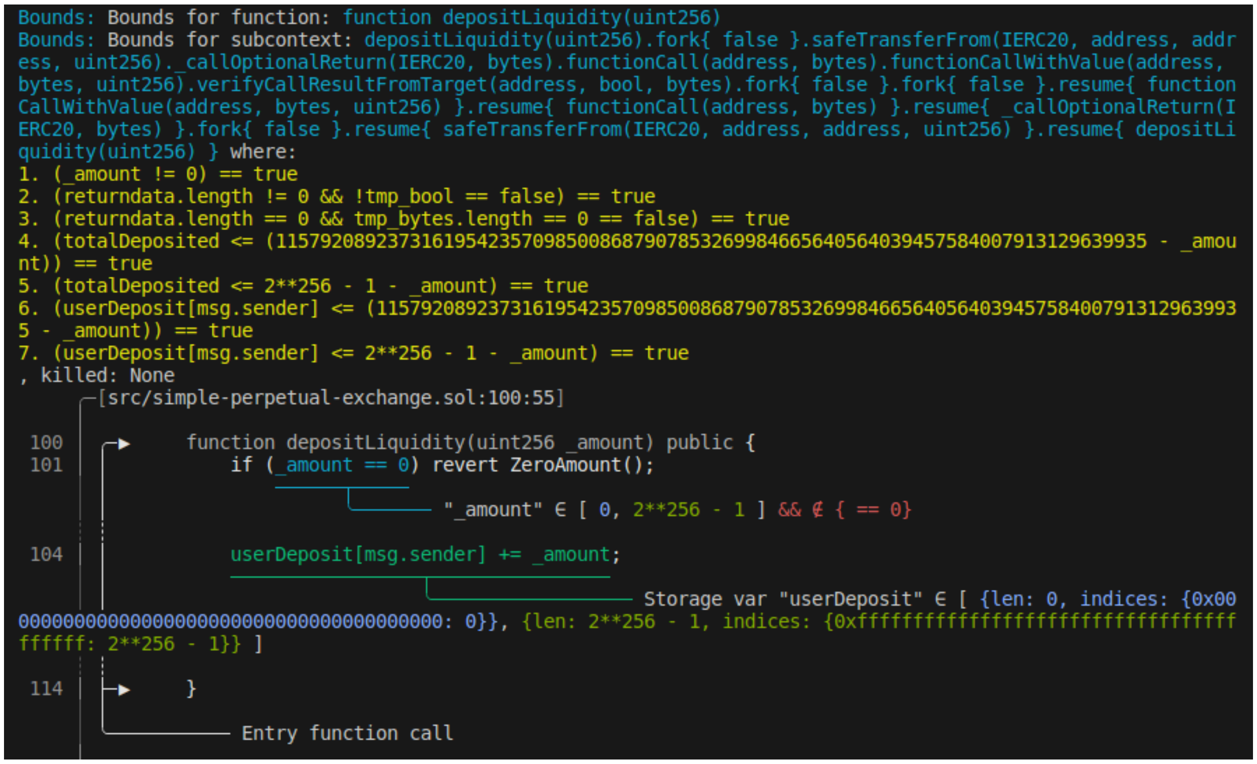
I used this command to initiate the test



```
pyrometer src/simple-perpetual-exchange.sol -vv --remappings ./remapping.txt
```

pyrometer - command function
src/simple-perpetual-exchange.sol - the target directory
-vv - the amount of verbosity (medium)
–remappings ./remapping.txt - to specify remappings for forge

** I will be not be showing any SafeERC20 external library calls
** I will only be showing three results to reduce repetitiveness in this report, outputs for depositLiquidity(), withdrawLiquidity(), and openPosition()

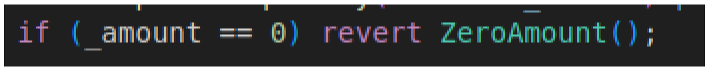


```
Bounds: Bounds for function: function depositLiquidity(uint256)
Bounds: Bounds for subcontext: depositLiquidity(uint256).fork{ false }.safeTransferFrom(IERC20, address, addr
ess, uint256)._callOptionalReturn(IERC20, bytes).functionCall(address, bytes).functionCallWithValue(address,
bytes, uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false }.fork{ false }.resume{ function
CallWithValue(address, bytes, uint256) }.resume{ functionCall(address, bytes) }.resume{ _callOptionalReturn(I
ERC20, bytes) }.fork{ false }.resume{ safeTransferFrom(IERC20, address, address, uint256) }.resume{ depositLi
quidity(uint256) } where:
1. (_amount != 0) == true
2. (returndata.length != 0 && !tmp_bool == false) == true
3. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
4. (totalDeposited <= (115792089237316195423570985008687907853269984665640564039457584007913129639935 - _amou
nt)) == true
5. (totalDeposited <= 2**256 - 1 - _amount) == true
6. (userDeposit[msg.sender] <= (11579208923731619542357098500868790785326998466564056403945758400791312963993
5 - _amount)) == true
7. (userDeposit[msg.sender] <= 2**256 - 1 - _amount) == true
, killed: None
      ┌─[src/simple-perpetual-exchange.sol:100:55]
      │
  100 │  ┌─▶     function depositLiquidity(uint256 _amount) public {
  101 │  │           if (_amount == 0) revert ZeroAmount();
      │  │               ────────────
      │  │                     └────────── "_amount" ∈ [ 0, 2**256 - 1 ] && ∉ { == 0}
      ┆  ┆
  104 │  │           userDeposit[msg.sender] += _amount;
      │  │           ───────────────────────────────────
      │  │                          └──────────────────── Storage var "userDeposit" ∈ [ {len: 0, indices: {0x00
00000000000000000000000000000000000000: 0}}, {len: 2**256 - 1, indices: {0xffffffffffffffffffffffffffffffffff
ffffff: 2**256 - 1}} ]
      ┆  ┆
  114 │  ├─▶     }
      │  │
      │  └──────────────── Entry function call
      │
```

Checking bounds for `depositLiquidity(uint256)`

In the yellow ordered text, it lists constraints for the function.

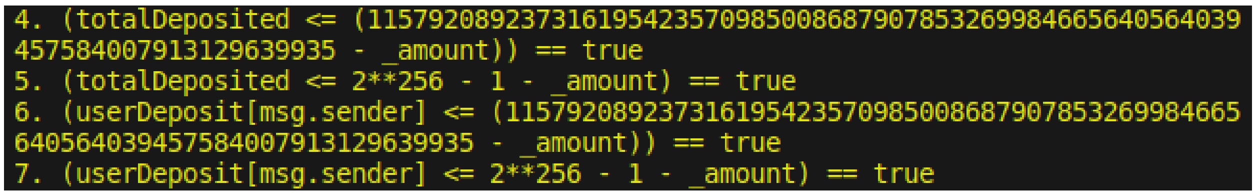
examples: `if ( _amount != 0) == true`
The argument that that is passed into the _amount parameter must not be zero

In the actual function code, it is this: 

```
if (_amount == 0) revert ZeroAmount();
```

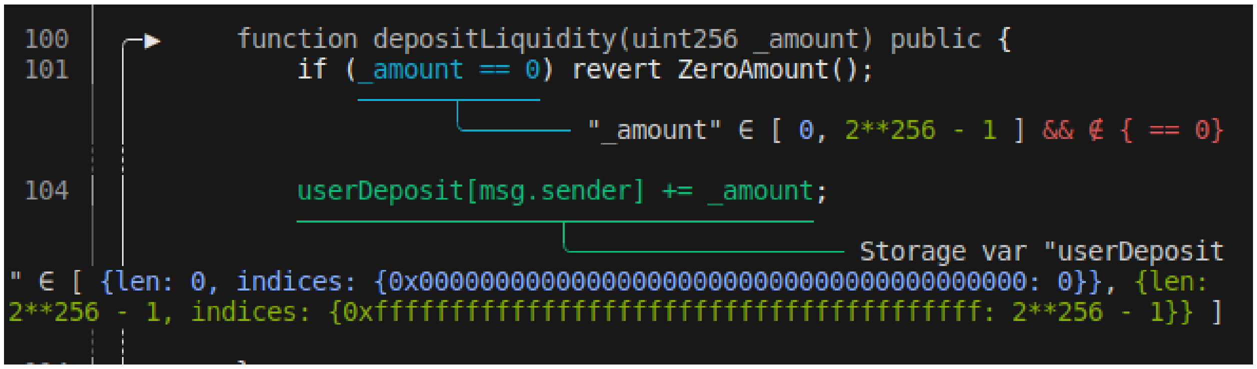
The pyrometer output seems to list expressions that persist in the code, including if statements that can cause a revert if true.

These next constraints (4 through 7) mark that these values must be less than or equal to 2**256 - 1, which is the maximum value that can be assigned to a uint256



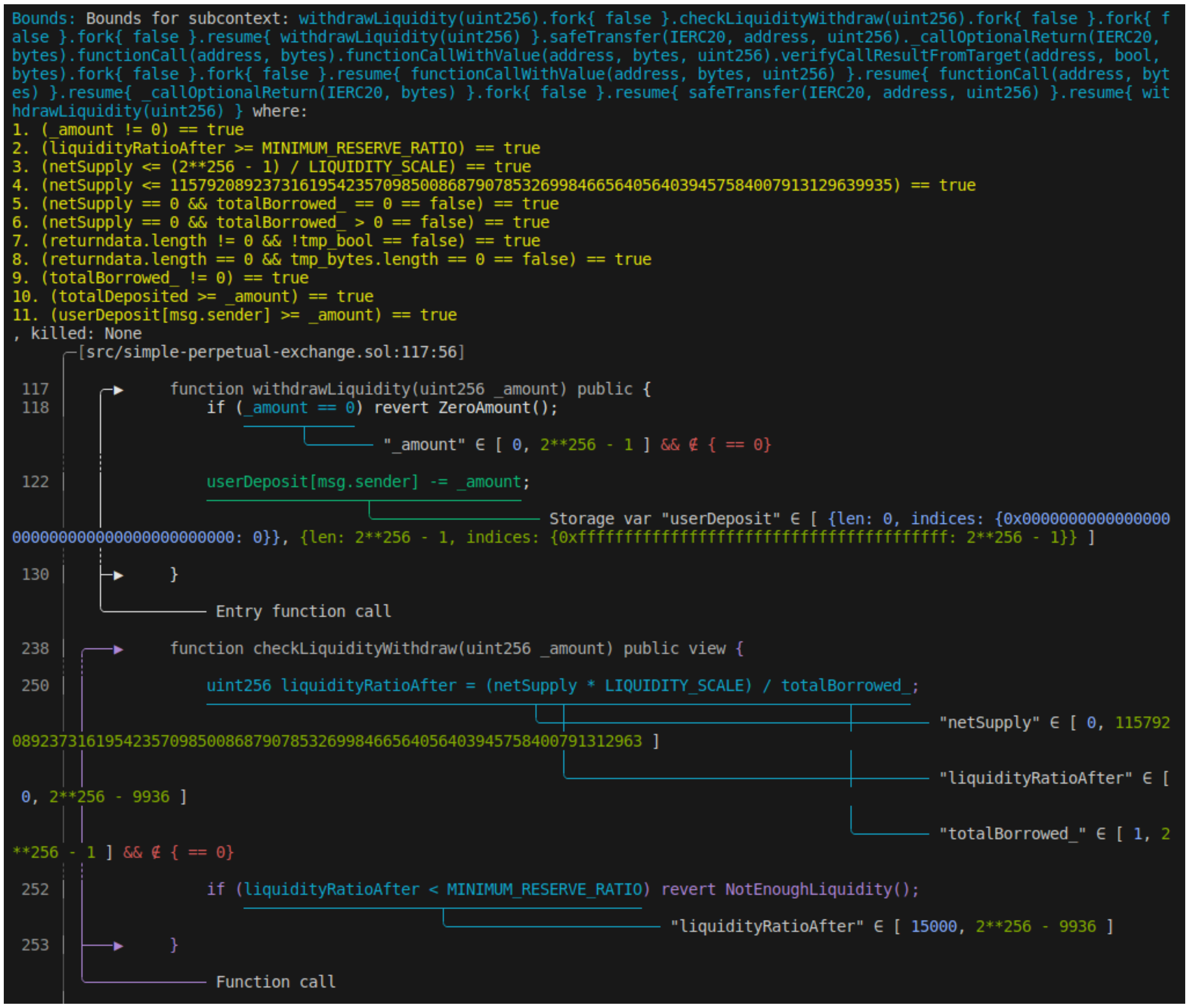
```
4. (totalDeposited <= (115792089237316195423570985008687907853269984665640564039
457584007913129639935 - _amount)) == true
5. (totalDeposited <= 2**256 - 1 - _amount) == true
6. (userDeposit[msg.sender] <= (115792089237316195423570985008687907853269984665
640564039457584007913129639935 - _amount)) == true
7. (userDeposit[msg.sender] <= 2**256 - 1 - _amount) == true
```

The _amount parameter, it's function can be an element of (∈) 0 and 2**256 - 1 and the element cannot be equal to zero.
The analysis of the storage mapping userDeposit[] indicates that the following description is about the possible states.
`{len: 0, indices: {0x0000000000000000000000000000000000000000: 0}}`
suggests a state of the mapping where it's essentially empty. This might represent the initial state of the mapping when no deposits have been made. The 0x000...000 address (a 40-hex-character string) represents the zero address in Ethereum, often used as a default or placeholder.
`{len: 2**256 - 1, indices:`
`{0xffffffffffffffffffffffffffffffffffffffff: 2**256 - 1}}` represents a theoretical maximum state of the mapping, where it can have up to 2**256 - 1 different entries (which is an enormously large number, essentially the maximum number of unique addresses possible in Ethereum). The 0xfff...fff address is the last possible address in the Ethereum address space.



```
    ╭─▶     function depositLiquidity(uint256 _amount) public {
    │           if (_amount == 0) revert ZeroAmount();
    │               ────────┬────────
    │                       ╰──────── "_amount" ∈ [ 0, 2**256 - 1 ] && ∉ { == 0}
    ┆
    │           userDeposit[msg.sender] += _amount;
    │           ─────────────────┬─────────────────
    │                            ╰──────────────── Storage var "userDeposit
" ∈ [ {len: 0, indices: {0x0000000000000000000000000000000000000000: 0}}, {len:
2**256 - 1, indices: {0xffffffffffffffffffffffffffffffffffffffff: 2**256 - 1}} ]
    ┆
```

Next the tool runs outputs results for withdrawLiquidity()



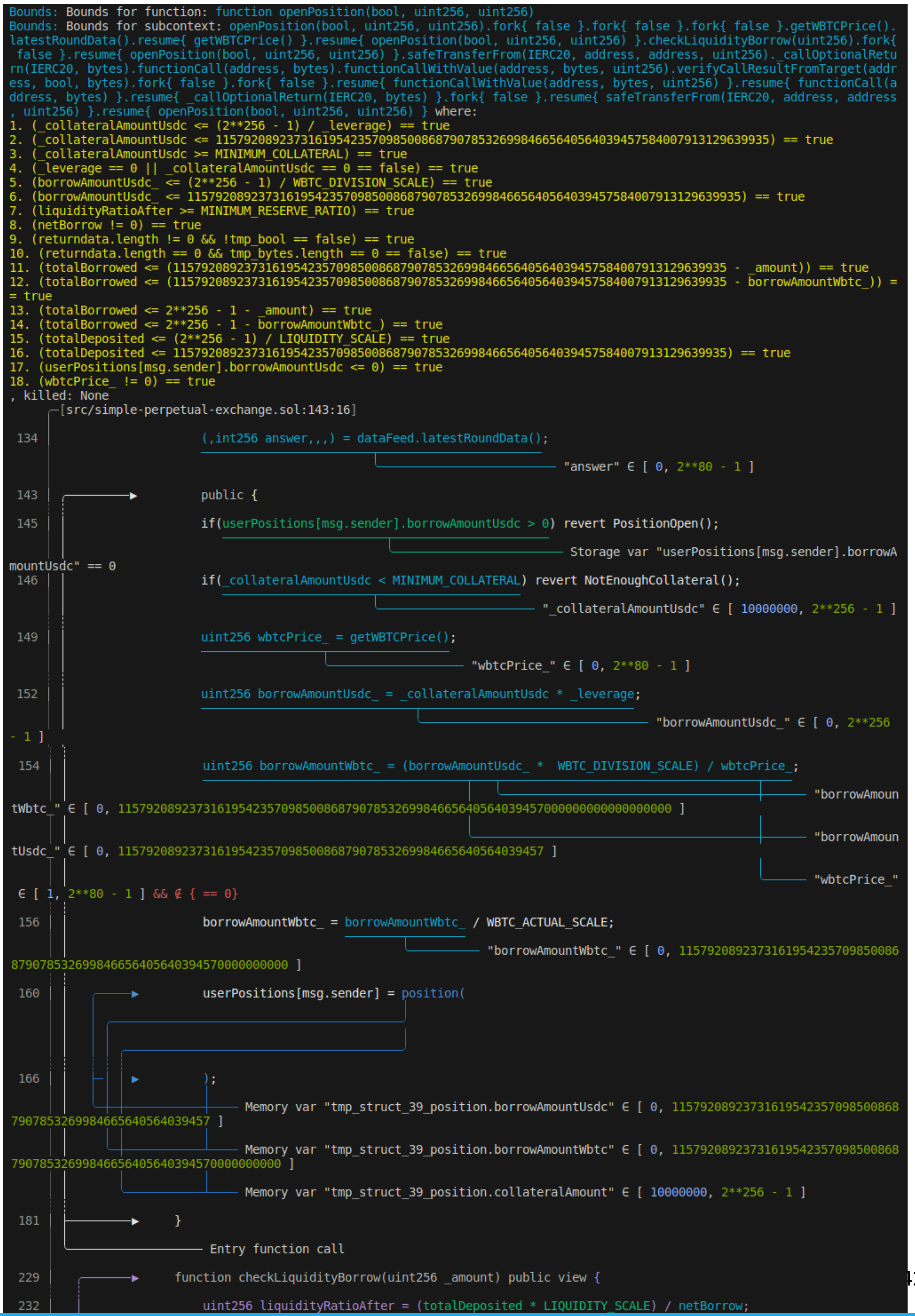
```
Bounds: Bounds for subcontext: withdrawLiquidity(uint256).fork{ false }.checkLiquidityWithdraw(uint256).fork{ false }.fork{ f
alse }.fork{ false }.resume{ withdrawLiquidity(uint256) }.safeTransfer(IERC20, address, uint256)._callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes, uint256).verifyCallResultFromTarget(address, bool,
bytes).fork{ false }.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256) }.resume{ functionCall(address, byt
es) }.resume{ _callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{ safeTransfer(IERC20, address, uint256) }.resume{ wit
hdrawLiquidity(uint256) } where:
1. (_amount != 0) == true
2. (liquidityRatioAfter >= MINIMUM_RESERVE_RATIO) == true
3. (netSupply <= (2**256 - 1) / LIQUIDITY_SCALE) == true
4. (netSupply <= 115792089237316195423570985008687907853269984665640564039457584007913129639935) == true
5. (netSupply == 0 && totalBorrowed_ == 0 == false) == true
6. (netSupply == 0 && totalBorrowed_ > 0 == false) == true
7. (returndata.length != 0 && !tmp_bool == false) == true
8. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
9. (totalBorrowed_ != 0) == true
10. (totalDeposited >= _amount) == true
11. (userDeposit[msg.sender] >= _amount) == true
, killed: None
      ╭─[src/simple-perpetual-exchange.sol:117:56]
      │
      │  ╭─▶     function withdrawLiquidity(uint256 _amount) public {
      │  │           if (_amount == 0) revert ZeroAmount();
      │  │               ──────┬──────
      │  │                     ╰─────── "_amount" ∈ [ 0, 2**256 - 1 ] && ∉ { == 0}
      │  ┆
      │  │           userDeposit[msg.sender] -= _amount;
      │  │           ─────────────────┬─────────────────
      │  │                            ╰─────────────── Storage var "userDeposit" ∈ [ {len: 0, indices: {0x0000000000000000
000000000000000000000000: 0}}, {len: 2**256 - 1, indices: {0xffffffffffffffffffffffffffffffffffffffff: 2**256 - 1}} ]
      │  ┆
      │  ├─▶     }
      │  │
      │  ╰──────────── Entry function call
      │
      │ ╭──▶     function checkLiquidityWithdraw(uint256 _amount) public view {
      │ ┆
      │ │            uint256 liquidityRatioAfter = (netSupply * LIQUIDITY_SCALE) / totalBorrowed_;
      │ │            ──────────────────────────────┬─┬──────────────────────────┬───
      │ │                                          ╰─┼──────────────────────────┼────────── "netSupply" ∈ [ 0, 115792
089237316195423570985008687907853269984665640564039457584007913129639935 ]
      │ │                                            │                          │
      │ │                                            ╰──────────────────────────┼────────── "liquidityRatioAfter" ∈ [
 0, 2**256 - 9936 ]
      │ │                                                                       │
      │ │                                                                       ╰────────── "totalBorrowed_" ∈ [ 1, 2
**256 - 1 ] && ∉ { == 0}
      │ │
      │ │            if (liquidityRatioAfter < MINIMUM_RESERVE_RATIO) revert NotEnoughLiquidity();
      │ │                ────────────────────┬────────────────────
      │ │                                    ╰──────────────────── "liquidityRatioAfter" ∈ [ 15000, 2**256 - 9936 ]
      │ │
      │ ├──▶     }
      │ │
      │ ╰──────────── Function call
      │
```

The constraints listed are

1. The _amount parameter must not be 0.
2. The liquidityRatioAfter (the ratio of deposits vs borrows) must be higher than the MINIMUM_RESERVE_RATIO
3. The net supply, used to calculate the liquidityRatioAfter must be less than or equal to 2**256 - 1, which is the maximum value for uint256, divided by the LIQUIDITY_SCALE, a constant whose purpose is to properly scale the calculation that results in the minimum
4. States netSupply should be less than or equal to the maximum uint256 value
5. In the checkLiquidityWithdraw function, states that the netSupply and totalBorrowed must be true (a condition I set the function to return for to prevent dividing by zero and causing an error)
6. In the checkLiquidityWithdraw function, states that the netSupply and totalBorrowed must be true states that the netSupply must be zero and the totalBorrowed must be greater than 0 (a condition I set to throw an error for the case of a user withdrawing all of the liquidity in the pool and there are active loans, to prevent a division by zero error)
7. Skipped because it is in an external library for SafeERC20
8. Skipped because it is in an external library for SafeERC20
9. States totalBorrowed must not be 0 (I'm not sure why it outputs this)
10. totalDeposit must be > 0 (I'm not sure why it outputs this)
11. userDeposit must be > 0 (not sure why it outputs this)

** ranges of each variable and calculation omitted

Next the tool runs outputs results for openPosition() shown in the following page



```
Bounds: Bounds for function: function openPosition(bool, uint256, uint256)
Bounds: Bounds for subcontext: openPosition(bool, uint256, uint256).fork{ false }.fork{ false }.fork{ false }.getWBTCPrice().
latestRoundData().resume{ getWBTCPrice() }.resume{ openPosition(bool, uint256, uint256) }.checkLiquidityBorrow(uint256).fork{
 false }.resume{ openPosition(bool, uint256, uint256) }.safeTransferFrom(IERC20, address, address, uint256)._callOptionalRetu
rn(IERC20, bytes).functionCall(address, bytes).functionCallWithValue(address, bytes, uint256).verifyCallResultFromTarget(addr
ess, bool, bytes).fork{ false }.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256) }.resume{ functionCall(a
ddress, bytes) }.resume{ _callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{ safeTransferFrom(IERC20, address, address
, uint256) }.resume{ openPosition(bool, uint256, uint256) } where:
1. (_collateralAmountUsdc <= (2**256 - 1) / _leverage) == true
2. (_collateralAmountUsdc <= 115792089237316195423570985008687907853269984665640564039457584007913129639935) == true
3. (_collateralAmountUsdc >= MINIMUM_COLLATERAL) == true
4. (_leverage == 0 || _collateralAmountUsdc == 0 == false) == true
5. (borrowAmountUsdc_ <= (2**256 - 1) / WBTC_DIVISION_SCALE) == true
6. (borrowAmountUsdc_ <= 115792089237316195423570985008687907853269984665640564039457584007913129639935) == true
7. (liquidityRatioAfter >= MINIMUM_RESERVE_RATIO) == true
8. (netBorrow != 0) == true
9. (returndata.length != 0 && !tmp_bool == false) == true
10. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
11. (totalBorrowed <= (115792089237316195423570985008687907853269984665640564039457584007913129639935 - _amount)) == true
12. (totalBorrowed <= (115792089237316195423570985008687907853269984665640564039457584007913129639935 - borrowAmountWbtc_)) =
= true
13. (totalBorrowed <= 2**256 - 1 - _amount) == true
14. (totalBorrowed <= 2**256 - 1 - borrowAmountWbtc_) == true
15. (totalDeposited <= (2**256 - 1) / LIQUIDITY_SCALE) == true
16. (totalDeposited <= 115792089237316195423570985008687907853269984665640564039457584007913129639935) == true
17. (userPositions[msg.sender].borrowAmountUsdc <= 0) == true
18. (wbtcPrice_ != 0) == true
, killed: None
      ┌─[src/simple-perpetual-exchange.sol:143:16]
      │
 134  │                     (,int256 answer,,,) = dataFeed.latestRoundData();
      │                     ───────────────────────┬─────────────────────────
      │                                            └───────────────── "answer" ∈ [ 0, 2**80 - 1 ]
      │
 143  │ ┌──────────▶          public {
 145  │ │                     if(userPositions[msg.sender].borrowAmountUsdc > 0) revert PositionOpen();
      │ │                       ─────────────────────┬──────────────────────
      │ │                                            └─────────────── Storage var "userPositions[msg.sender].borrowA
mountUsdc" == 0
 146  │ │                     if(_collateralAmountUsdc < MINIMUM_COLLATERAL) revert NotEnoughCollateral();
      │ │                       ─────────────────┬─────────────────────
      │ │                                        └─────────────── "_collateralAmountUsdc" ∈ [ 10000000, 2**256 - 1 ]
      │ │
 149  │ │                     uint256 wbtcPrice_ = getWBTCPrice();
      │ │                     ─────────────┬──────────────────
      │ │                                  └──────────────── "wbtcPrice_" ∈ [ 0, 2**80 - 1 ]
      │ │
 152  │ │                     uint256 borrowAmountUsdc_ = _collateralAmountUsdc * _leverage;
      │ │                     ──────────────────┬──────────────────────────────────
      │ │                                       └──────────────── "borrowAmountUsdc_" ∈ [ 0, 2**256
- 1 ]
 154  │ │                     uint256 borrowAmountWbtc_ = (borrowAmountUsdc_ *  WBTC_DIVISION_SCALE) / wbtcPrice_;
      │ │                     ──────────────────────┬───┬─────────────────────────┬──────────
      │ │                                           │   └─────────────────────────┼──────── "borrowAmoun
tWbtc_" ∈ [ 0, 115792089237316195423570985008687907853269984665640564039457000000000000000000 ]
      │ │                                           │                             │
      │ │                                           └─────────────────────────────┼──────── "borrowAmoun
tUsdc_" ∈ [ 0, 115792089237316195423570985008687907853269984665640564039457 ]
      │ │                                                                         │
      │ │                                                                         └──────── "wbtcPrice_"
 ∈ [ 1, 2**80 - 1 ] && ∉ { == 0}
 156  │ │                     borrowAmountWbtc_ = borrowAmountWbtc_ / WBTC_ACTUAL_SCALE;
      │ │                                         ────────┬────────
      │ │                                                 └──────────── "borrowAmountWbtc_" ∈ [ 0, 115792089237316195423570985008
6
879078532699846656405640394570000000000 ]
 160  │ │     ┌──────────▶          userPositions[msg.sender] = position(
      │ │     │  ┌──────────────────────────────────────────────────┘
      │ │     │  │  ┌───────────────────────────────────────────────┘
 166  │ │     ├─ │  │   ▶          );
      │ │     └──┼──┼──────────────┴──── Memory var "tmp_struct_39_position.borrowAmountUsdc" ∈ [ 0, 115792089237316195423570985008
68
79078532699846656405640564039457 ]
      │ │        └──┼──────────────┴──── Memory var "tmp_struct_39_position.borrowAmountWbtc" ∈ [ 0, 115792089237316195423570985008
68
790785326998466564056403945700000000000 ]
      │ │           └──────────────┴──── Memory var "tmp_struct_39_position.collateralAmount" ∈ [ 10000000, 2**256 - 1 ]
      │ │
 181  │ ├──────────▶      }
      │ │
      │ └─────────────────────── Entry function call
      │
 229  │   ┌──────────▶      function checkLiquidityBorrow(uint256 _amount) public view {
 232  │   │                     uint256 liquidityRatioAfter = (totalDeposited * LIQUIDITY_SCALE) / netBorrow;
```

Constraints:

1. The _collateralAmountUsdc parameter argument must be less than or equal to the maximum uint256 value / _leverage. (I assume this is the reciprocal operation of _collateralAmountUsdc * _leverage)
2. The _collateralAmountUsdc parameter argument must be less than or equal to the maximum uint256 value
3. The _collateralAmountUsdc parameter argument must be greater than or equal to MINIMUM_COLLATERAL (10**6 or $10 in USDC)
4. The _leverage, _collateralAmountUsdc arguments must not be equal to zero
5. The borrowAmountUsdc final calculation variable must be less than or equal to the maximum uin256 divided by the WBTC_DIVISION_SCALE constant which is used to properly scale integer mathematical operations.
6. The borrowAmountUsdc variable must be less than or equal to the maximum uint256 value
7. The new liquidity ratio must be greater than or equal to the minimum allowed amount
8. netBorrow must not equal zero (Presumed for the checkLiquidity() function)
9. Omitted because it's apart of the SafeERC20 Library
10. Omitted because it's apart of the SafeERC20 Library
11. totalBorrowed must be less than or equal to the maximum uint256 value minus the total amount of usdc collateral (Not sure why it outputs this)
12. totalBorrowed must be less than or equal to the maximum uint256 value minus the borrowed bitcoin amount
13. Re iteration of constraint 11
14. Re iteration of constraint 12
15. totalDeposited must be less than or equal to the maximum uint256 value divided by the LIQUIDITY_SCALE
16. totalDeposited must be less than or equal to the maximum uint256
17. userDeposit must be less than or equal to zero (presumably enforced because I don't let people open a new position if they already have a position)
18. Bitcoin price must not be zero

** ranges of each variable and calculation omitted

Conclusion:

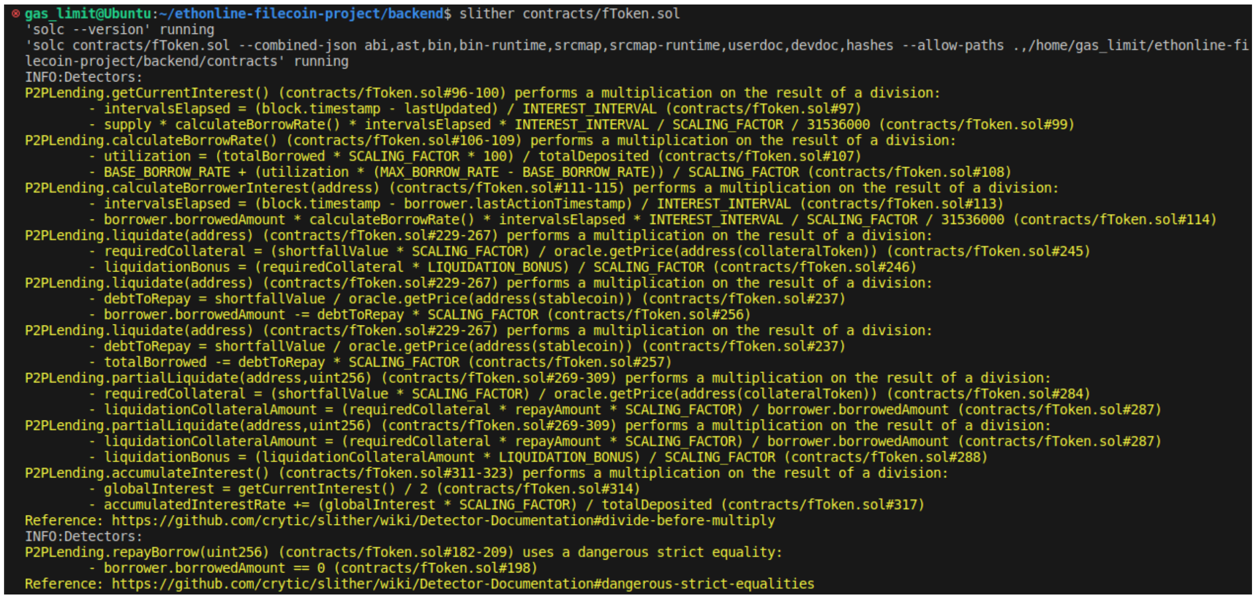
Pyrometer serves as a useful tool in smart contract analysis, particularly in its ability to monitor invariants—key conditions meant to remain constant throughout the contract's lifecycle. While this feature helps in maintaining the structural consistency of the contract, its effectiveness in complex scenarios may vary. Additionally, Pyrometer offers insights into the potential value ranges of each variable state, which is useful for understanding the limits and behaviors of these variables. However, the accuracy of these insights can depend on the specificities of the contract being analyzed.

Readings / Video content

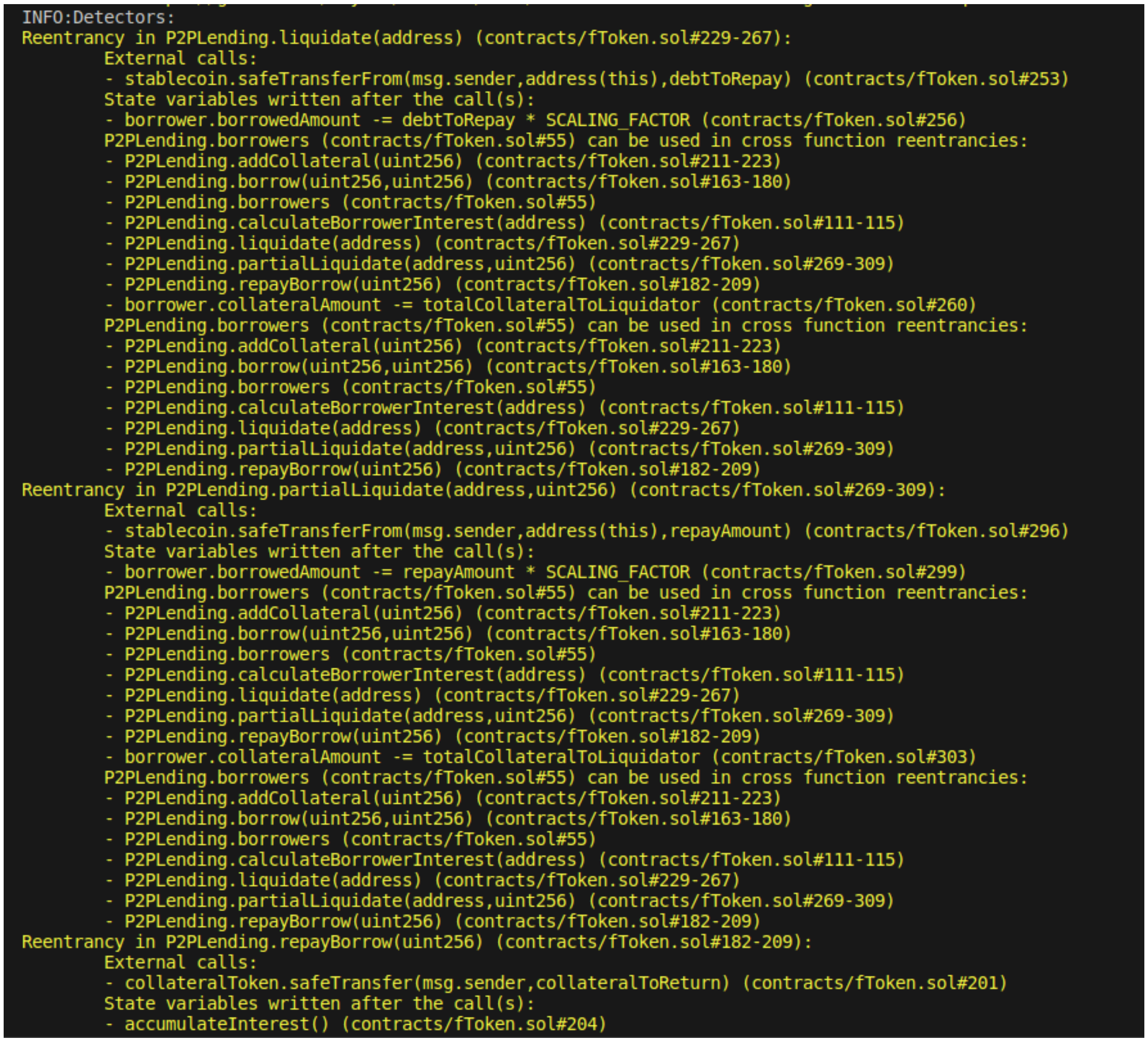
1. Owen Thurm - Advanced Web3 Security Course | Part 1 (11 hours)
   https://www.youtube.com/watch?v=DRZogmD647U
2. FileLend Project Repository
   https://github.com/JustUzair/ethonline-filecoin-project/blob/master/backend/contracts/fToken.sol
3. Perpetual Exchange Repository
   https://github.com/gas-limit/PURPetual-exchange.git
4.

Additional Screenshots

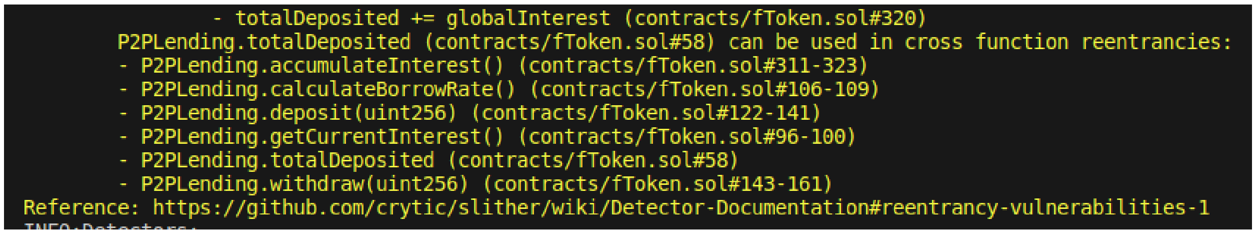
Slither test on lending contract



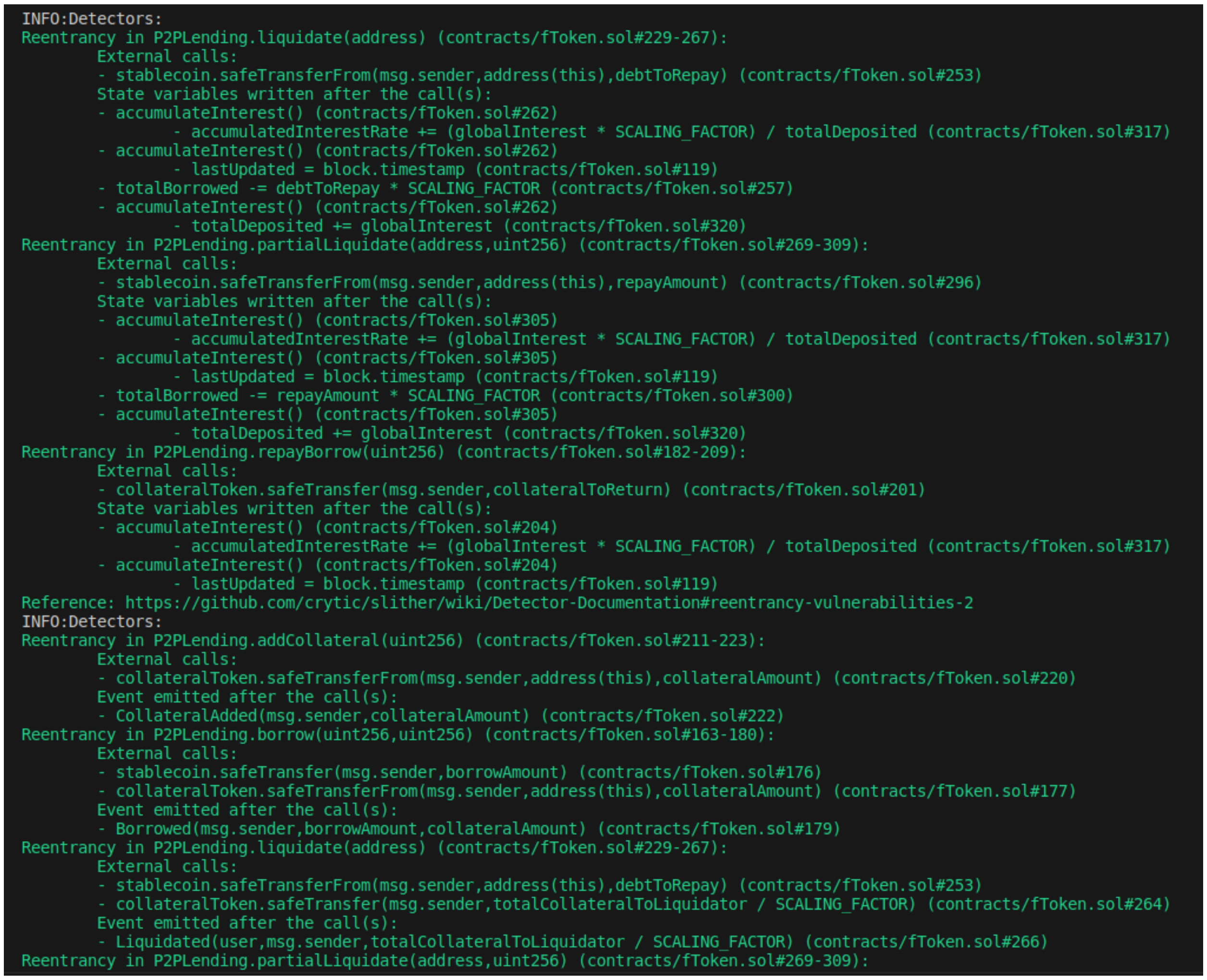
```
⊗ gas_limit@Ubuntu:~/ethonline-filecoin-project/backend$ slither contracts/fToken.sol
  'solc --version' running
  'solc contracts/fToken.sol --combined-json abi,ast,bin,bin-runtime,srcmap,srcmap-runtime,userdoc,devdoc,hashes --allow-paths .,/home/gas_limit/ethonline-fi
  lecoin-project/backend/contracts' running
  INFO:Detectors:
  P2PLending.getCurrentInterest() (contracts/fToken.sol#96-100) performs a multiplication on the result of a division:
          - intervalsElapsed = (block.timestamp - lastUpdated) / INTEREST_INTERVAL (contracts/fToken.sol#97)
          - supply * calculateBorrowRate() * intervalsElapsed * INTEREST_INTERVAL / SCALING_FACTOR / 31536000 (contracts/fToken.sol#99)
  P2PLending.calculateBorrowRate() (contracts/fToken.sol#106-109) performs a multiplication on the result of a division:
          - utilization = (totalBorrowed * SCALING_FACTOR * 100) / totalDeposited (contracts/fToken.sol#107)
          - BASE_BORROW_RATE + (utilization * (MAX_BORROW_RATE - BASE_BORROW_RATE)) / SCALING_FACTOR (contracts/fToken.sol#108)
  P2PLending.calculateBorrowerInterest(address) (contracts/fToken.sol#111-115) performs a multiplication on the result of a division:
          - intervalsElapsed = (block.timestamp - borrower.lastActionTimestamp) / INTEREST_INTERVAL (contracts/fToken.sol#113)
          - borrower.borrowedAmount * calculateBorrowRate() * intervalsElapsed * INTEREST_INTERVAL / SCALING_FACTOR / 31536000 (contracts/fToken.sol#114)
  P2PLending.liquidate(address) (contracts/fToken.sol#229-267) performs a multiplication on the result of a division:
          - requiredCollateral = (shortfallValue * SCALING_FACTOR) / oracle.getPrice(address(collateralToken)) (contracts/fToken.sol#245)
          - liquidationBonus = (requiredCollateral * LIQUIDATION_BONUS) / SCALING_FACTOR (contracts/fToken.sol#246)
  P2PLending.liquidate(address) (contracts/fToken.sol#229-267) performs a multiplication on the result of a division:
          - debtToRepay = shortfallValue / oracle.getPrice(address(stablecoin)) (contracts/fToken.sol#237)
          - borrower.borrowedAmount -= debtToRepay * SCALING_FACTOR (contracts/fToken.sol#256)
  P2PLending.liquidate(address) (contracts/fToken.sol#229-267) performs a multiplication on the result of a division:
          - debtToRepay = shortfallValue / oracle.getPrice(address(stablecoin)) (contracts/fToken.sol#237)
          - totalBorrowed -= debtToRepay * SCALING_FACTOR (contracts/fToken.sol#257)
  P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309) performs a multiplication on the result of a division:
          - requiredCollateral = (shortfallValue * SCALING_FACTOR) / oracle.getPrice(address(collateralToken)) (contracts/fToken.sol#284)
          - liquidationCollateralAmount = (requiredCollateral * repayAmount * SCALING_FACTOR) / borrower.borrowedAmount (contracts/fToken.sol#287)
  P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309) performs a multiplication on the result of a division:
          - liquidationCollateralAmount = (requiredCollateral * repayAmount * SCALING_FACTOR) / borrower.borrowedAmount (contracts/fToken.sol#287)
          - liquidationBonus = (liquidationCollateralAmount * LIQUIDATION_BONUS) / SCALING_FACTOR (contracts/fToken.sol#288)
  P2PLending.accumulateInterest() (contracts/fToken.sol#311-323) performs a multiplication on the result of a division:
          - globalInterest = getCurrentInterest() / 2 (contracts/fToken.sol#314)
          - accumulatedInterestRate += (globalInterest * SCALING_FACTOR) / totalDeposited (contracts/fToken.sol#317)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
  INFO:Detectors:
  P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209) uses a dangerous strict equality:
          - borrower.borrowedAmount == 0 (contracts/fToken.sol#198)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
```



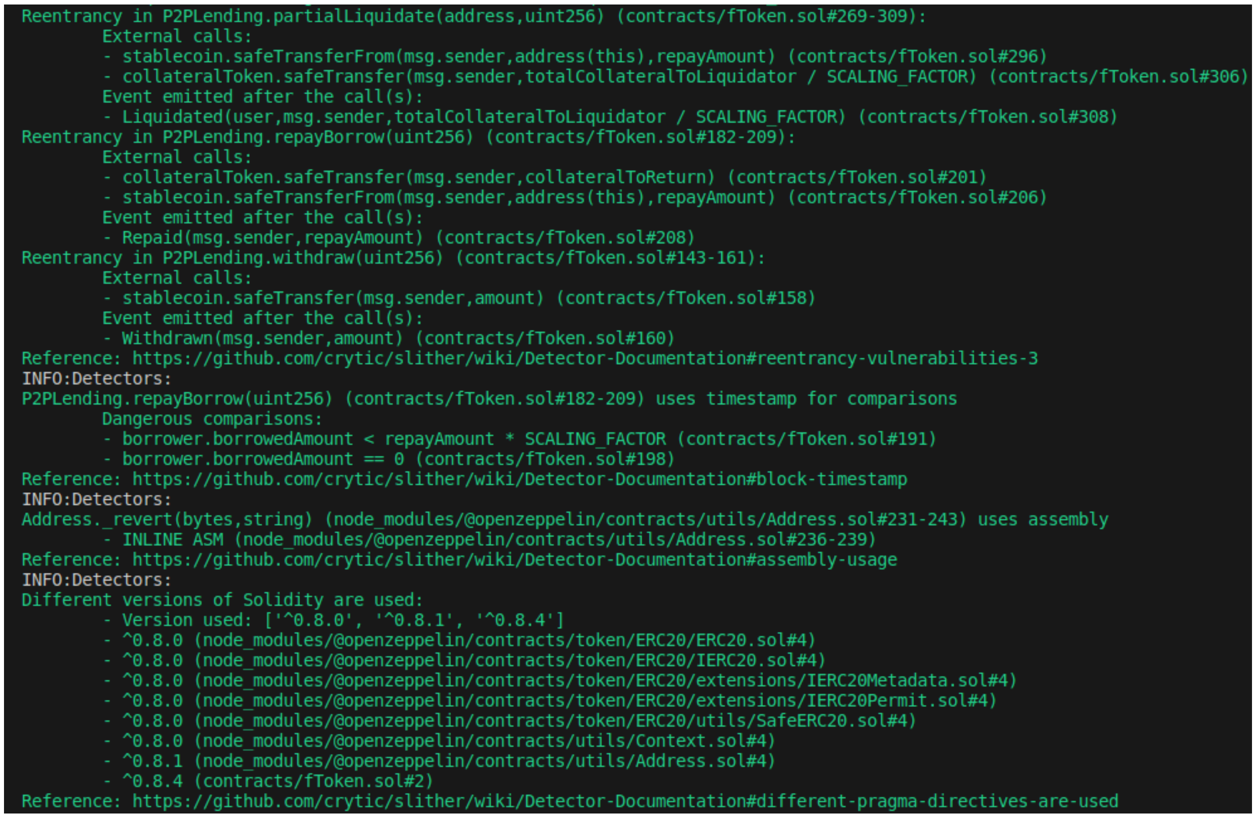
```
INFO:Detectors:
Reentrancy in P2PLending.liquidate(address) (contracts/fToken.sol#229-267):
	External calls:
	- stablecoin.safeTransferFrom(msg.sender,address(this),debtToRepay) (contracts/fToken.sol#253)
	State variables written after the call(s):
	- borrower.borrowedAmount -= debtToRepay * SCALING_FACTOR (contracts/fToken.sol#256)
	P2PLending.borrowers (contracts/fToken.sol#55) can be used in cross function reentrancies:
	- P2PLending.addCollateral(uint256) (contracts/fToken.sol#211-223)
	- P2PLending.borrow(uint256,uint256) (contracts/fToken.sol#163-180)
	- P2PLending.borrowers (contracts/fToken.sol#55)
	- P2PLending.calculateBorrowerInterest(address) (contracts/fToken.sol#111-115)
	- P2PLending.liquidate(address) (contracts/fToken.sol#229-267)
	- P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309)
	- P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209)
	- borrower.collateralAmount -= totalCollateralToLiquidator (contracts/fToken.sol#260)
	P2PLending.borrowers (contracts/fToken.sol#55) can be used in cross function reentrancies:
	- P2PLending.addCollateral(uint256) (contracts/fToken.sol#211-223)
	- P2PLending.borrow(uint256,uint256) (contracts/fToken.sol#163-180)
	- P2PLending.borrowers (contracts/fToken.sol#55)
	- P2PLending.calculateBorrowerInterest(address) (contracts/fToken.sol#111-115)
	- P2PLending.liquidate(address) (contracts/fToken.sol#229-267)
	- P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309)
	- P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209)
Reentrancy in P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309):
	External calls:
	- stablecoin.safeTransferFrom(msg.sender,address(this),repayAmount) (contracts/fToken.sol#296)
	State variables written after the call(s):
	- borrower.borrowedAmount -= repayAmount * SCALING_FACTOR (contracts/fToken.sol#299)
	P2PLending.borrowers (contracts/fToken.sol#55) can be used in cross function reentrancies:
	- P2PLending.addCollateral(uint256) (contracts/fToken.sol#211-223)
	- P2PLending.borrow(uint256,uint256) (contracts/fToken.sol#163-180)
	- P2PLending.borrowers (contracts/fToken.sol#55)
	- P2PLending.calculateBorrowerInterest(address) (contracts/fToken.sol#111-115)
	- P2PLending.liquidate(address) (contracts/fToken.sol#229-267)
	- P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309)
	- P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209)
	- borrower.collateralAmount -= totalCollateralToLiquidator (contracts/fToken.sol#303)
	P2PLending.borrowers (contracts/fToken.sol#55) can be used in cross function reentrancies:
	- P2PLending.addCollateral(uint256) (contracts/fToken.sol#211-223)
	- P2PLending.borrow(uint256,uint256) (contracts/fToken.sol#163-180)
	- P2PLending.borrowers (contracts/fToken.sol#55)
	- P2PLending.calculateBorrowerInterest(address) (contracts/fToken.sol#111-115)
	- P2PLending.liquidate(address) (contracts/fToken.sol#229-267)
	- P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309)
	- P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209)
Reentrancy in P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209):
	External calls:
	- collateralToken.safeTransfer(msg.sender,collateralToReturn) (contracts/fToken.sol#201)
	State variables written after the call(s):
	- accumulateInterest() (contracts/fToken.sol#204)
```



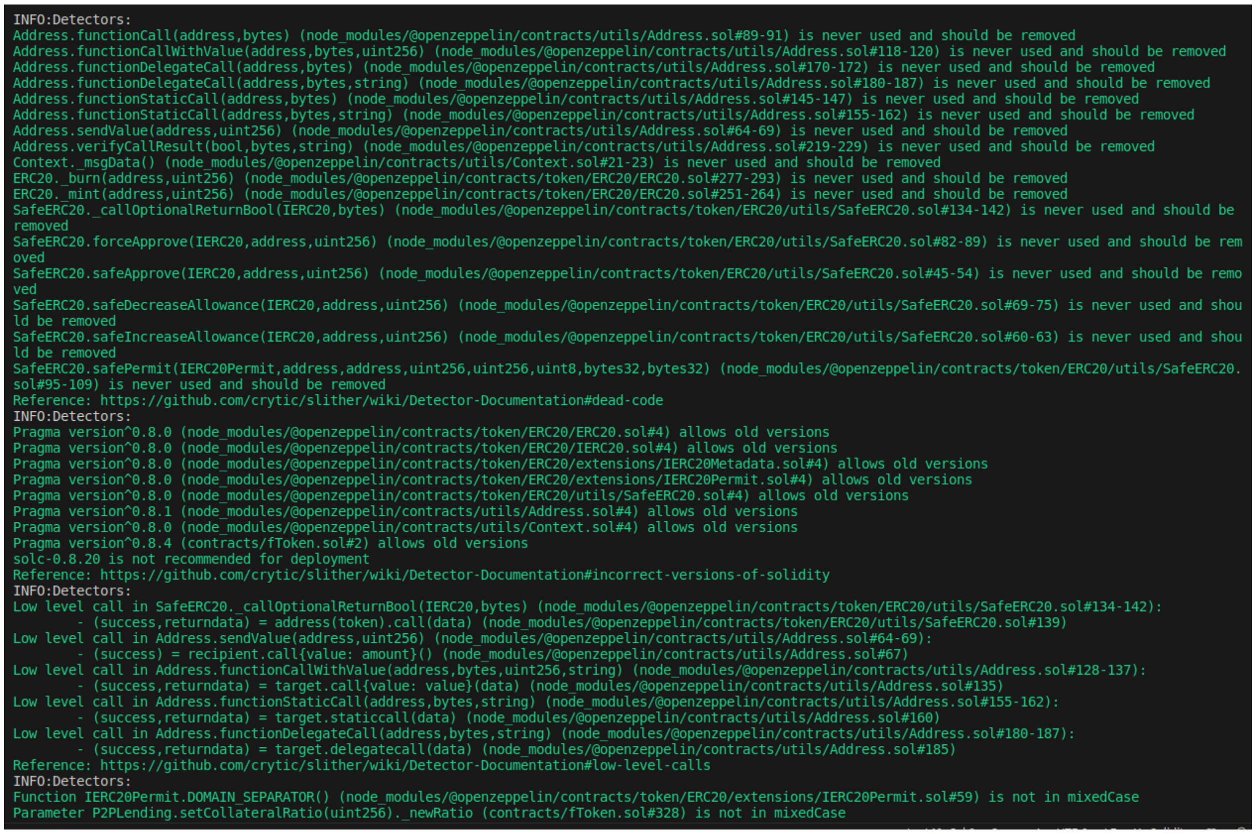
```
		- totalDeposited += globalInterest (contracts/fToken.sol#320)
	P2PLending.totalDeposited (contracts/fToken.sol#58) can be used in cross function reentrancies:
	- P2PLending.accumulateInterest() (contracts/fToken.sol#311-323)
	- P2PLending.calculateBorrowRate() (contracts/fToken.sol#106-109)
	- P2PLending.deposit(uint256) (contracts/fToken.sol#122-141)
	- P2PLending.getCurrentInterest() (contracts/fToken.sol#96-100)
	- P2PLending.totalDeposited (contracts/fToken.sol#58)
	- P2PLending.withdraw(uint256) (contracts/fToken.sol#143-161)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1
```



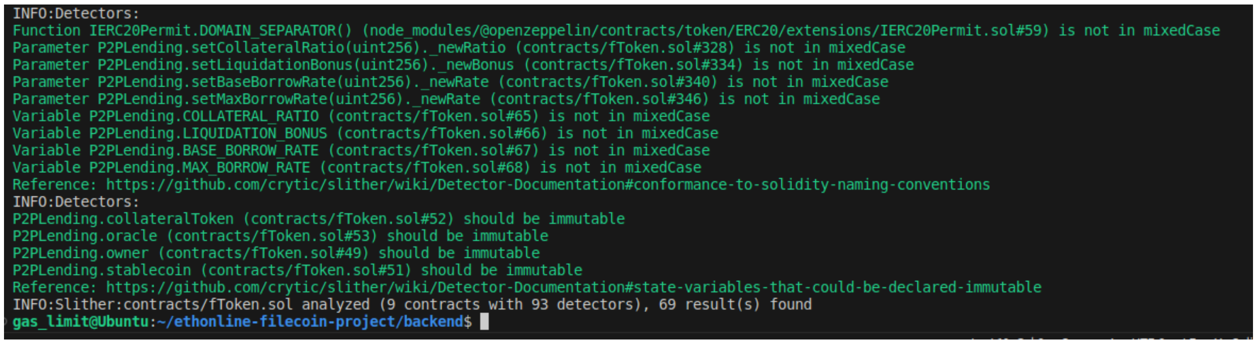
```
INFO:Detectors:
Reentrancy in P2PLending.liquidate(address) (contracts/fToken.sol#229-267):
        External calls:
        - stablecoin.safeTransferFrom(msg.sender,address(this),debtToRepay) (contracts/fToken.sol#253)
        State variables written after the call(s):
        - accumulateInterest() (contracts/fToken.sol#262)
                - accumulatedInterestRate += (globalInterest * SCALING_FACTOR) / totalDeposited (contracts/fToken.sol#317)
        - accumulateInterest() (contracts/fToken.sol#262)
                - lastUpdated = block.timestamp (contracts/fToken.sol#119)
        - totalBorrowed -= debtToRepay * SCALING_FACTOR (contracts/fToken.sol#257)
        - accumulateInterest() (contracts/fToken.sol#262)
                - totalDeposited += globalInterest (contracts/fToken.sol#320)
Reentrancy in P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309):
        External calls:
        - stablecoin.safeTransferFrom(msg.sender,address(this),repayAmount) (contracts/fToken.sol#296)
        State variables written after the call(s):
        - accumulateInterest() (contracts/fToken.sol#305)
                - accumulatedInterestRate += (globalInterest * SCALING_FACTOR) / totalDeposited (contracts/fToken.sol#317)
        - accumulateInterest() (contracts/fToken.sol#305)
                - lastUpdated = block.timestamp (contracts/fToken.sol#119)
        - totalBorrowed -= repayAmount * SCALING_FACTOR (contracts/fToken.sol#300)
        - accumulateInterest() (contracts/fToken.sol#305)
                - totalDeposited += globalInterest (contracts/fToken.sol#320)
Reentrancy in P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209):
        External calls:
        - collateralToken.safeTransfer(msg.sender,collateralToReturn) (contracts/fToken.sol#201)
        State variables written after the call(s):
        - accumulateInterest() (contracts/fToken.sol#204)
                - accumulatedInterestRate += (globalInterest * SCALING_FACTOR) / totalDeposited (contracts/fToken.sol#317)
        - accumulateInterest() (contracts/fToken.sol#204)
                - lastUpdated = block.timestamp (contracts/fToken.sol#119)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in P2PLending.addCollateral(uint256) (contracts/fToken.sol#211-223):
        External calls:
        - collateralToken.safeTransferFrom(msg.sender,address(this),collateralAmount) (contracts/fToken.sol#220)
        Event emitted after the call(s):
        - CollateralAdded(msg.sender,collateralAmount) (contracts/fToken.sol#222)
Reentrancy in P2PLending.borrow(uint256,uint256) (contracts/fToken.sol#163-180):
        External calls:
        - stablecoin.safeTransfer(msg.sender,borrowAmount) (contracts/fToken.sol#176)
        - collateralToken.safeTransferFrom(msg.sender,address(this),collateralAmount) (contracts/fToken.sol#177)
        Event emitted after the call(s):
        - Borrowed(msg.sender,borrowAmount,collateralAmount) (contracts/fToken.sol#179)
Reentrancy in P2PLending.liquidate(address) (contracts/fToken.sol#229-267):
        External calls:
        - stablecoin.safeTransferFrom(msg.sender,address(this),debtToRepay) (contracts/fToken.sol#253)
        - collateralToken.safeTransfer(msg.sender,totalCollateralToLiquidator / SCALING_FACTOR) (contracts/fToken.sol#264)
        Event emitted after the call(s):
        - Liquidated(user,msg.sender,totalCollateralToLiquidator / SCALING_FACTOR) (contracts/fToken.sol#266)
Reentrancy in P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309):
```



```
 Reentrancy in P2PLending.partialLiquidate(address,uint256) (contracts/fToken.sol#269-309):
        External calls:
        - stablecoin.safeTransferFrom(msg.sender,address(this),repayAmount) (contracts/fToken.sol#296)
        - collateralToken.safeTransfer(msg.sender,totalCollateralToLiquidator / SCALING_FACTOR) (contracts/fToken.sol#306)
        Event emitted after the call(s):
        - Liquidated(user,msg.sender,totalCollateralToLiquidator / SCALING_FACTOR) (contracts/fToken.sol#308)
 Reentrancy in P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209):
        External calls:
        - collateralToken.safeTransfer(msg.sender,collateralToReturn) (contracts/fToken.sol#201)
        - stablecoin.safeTransferFrom(msg.sender,address(this),repayAmount) (contracts/fToken.sol#206)
        Event emitted after the call(s):
        - Repaid(msg.sender,repayAmount) (contracts/fToken.sol#208)
 Reentrancy in P2PLending.withdraw(uint256) (contracts/fToken.sol#143-161):
        External calls:
        - stablecoin.safeTransfer(msg.sender,amount) (contracts/fToken.sol#158)
        Event emitted after the call(s):
        - Withdrawn(msg.sender,amount) (contracts/fToken.sol#160)
 Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
 INFO:Detectors:
 P2PLending.repayBorrow(uint256) (contracts/fToken.sol#182-209) uses timestamp for comparisons
        Dangerous comparisons:
        - borrower.borrowedAmount < repayAmount * SCALING_FACTOR (contracts/fToken.sol#191)
        - borrower.borrowedAmount == 0 (contracts/fToken.sol#198)
 Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
 INFO:Detectors:
 Address._revert(bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#231-243) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#236-239)
 Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
 INFO:Detectors:
 Different versions of Solidity are used:
        - Version used: ['^0.8.0', '^0.8.1', '^0.8.4']
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Permit.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
        - ^0.8.1 (node_modules/@openzeppelin/contracts/utils/Address.sol#4)
        - ^0.8.4 (contracts/fToken.sol#2)
 Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
```



```
INFO:Detectors:
Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#89-91) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#118-120) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#170-172) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#180-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#145-147) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#155-162) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#64-69) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#219-229) is never used and should be removed
Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#21-23) is never used and should be removed
ERC20._burn(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#277-293) is never used and should be removed
ERC20._mint(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#251-264) is never used and should be removed
SafeERC20._callOptionalReturnBool(IERC20,bytes) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#134-142) is never used and should be
removed
SafeERC20.forceApprove(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#82-89) is never used and should be rem
oved
SafeERC20.safeApprove(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#45-54) is never used and should be remo
ved
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#69-75) is never used and shou
ld be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#60-63) is never used and shou
ld be removed
SafeERC20.safePermit(IERC20Permit,address,address,uint256,uint256,uint8,bytes32,bytes32) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.
sol#95-109) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Permit.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#4) allows old versions
Pragma version^0.8.1 (node_modules/@openzeppelin/contracts/utils/Address.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) allows old versions
Pragma version^0.8.4 (contracts/fToken.sol#2) allows old versions
solc-0.8.20 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in SafeERC20._callOptionalReturnBool(IERC20,bytes) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#134-142):
        - (success,returndata) = address(token).call(data) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#139)
Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#64-69):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts/utils/Address.sol#67)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#128-137):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#135)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#155-162):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#160)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#180-187):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IERC20Permit.DOMAIN_SEPARATOR() (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Permit.sol#59) is not in mixedCase
Parameter P2PLending.setCollateralRatio(uint256)._newRatio (contracts/fToken.sol#328) is not in mixedCase
```



```
INFO:Detectors:
Function IERC20Permit.DOMAIN_SEPARATOR() (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Permit.sol#59) is not in mixedCase
Parameter P2PLending.setCollateralRatio(uint256)._newRatio (contracts/fToken.sol#328) is not in mixedCase
Parameter P2PLending.setLiquidationBonus(uint256)._newBonus (contracts/fToken.sol#334) is not in mixedCase
Parameter P2PLending.setBaseBorrowRate(uint256)._newRate (contracts/fToken.sol#340) is not in mixedCase
Parameter P2PLending.setMaxBorrowRate(uint256)._newRate (contracts/fToken.sol#346) is not in mixedCase
Variable P2PLending.COLLATERAL_RATIO (contracts/fToken.sol#65) is not in mixedCase
Variable P2PLending.LIQUIDATION_BONUS (contracts/fToken.sol#66) is not in mixedCase
Variable P2PLending.BASE_BORROW_RATE (contracts/fToken.sol#67) is not in mixedCase
Variable P2PLending.MAX_BORROW_RATE (contracts/fToken.sol#68) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
P2PLending.collateralToken (contracts/fToken.sol#52) should be immutable
P2PLending.oracle (contracts/fToken.sol#53) should be immutable
P2PLending.owner (contracts/fToken.sol#49) should be immutable
P2PLending.stablecoin (contracts/fToken.sol#51) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:contracts/fToken.sol analyzed (9 contracts with 93 detectors), 69 result(s) found
gas_limit@Ubuntu:~/ethonline-filecoin-project/backend$
```

Pyrometer Full Analysis

```
Bounds: Bounds for function: function sendValue(address, uint256)
Bounds: Bounds for subcontext: sendValue(address, uint256).fork{ false
}.fork{ false } where:
1. (tmp_uint256 >= amount) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:41:76]
    │
 41 │ ╭─▶     function sendValue(address payable recipient, uint256
amount) internal {
    ┆ ┆
 50 │ ├─▶     }
    │ │
    │ ╰───────────────────── Entry function call
────╯

Bounds: Bounds for function: function functionCall(address, bytes)
Bounds: Bounds for subcontext: functionCall(address,
bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) } where:
1. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:70:94]
```

```
     │
  70 │    ╭──▶      function functionCall(address target, bytes memory data)
internal returns (bytes memory) {
  71 │    │             return functionCallWithValue(target, data, 0);
     │    │
──────────────────────────────────────────────────────────────────
     │    │
 ╰─────────────────────────────────────────────────── returns:
"functionCallWithValue(address, bytes, uint256).0" ∈ [ {len: 0, indices:
{}}, {len: 2**256 - 1, indices: {}} ]
  72 │    ├──▶      }
     │    │
     │    ╰─────────────────────── Entry function call
     │
 118 │  ╭─────▶     ) internal view returns (bytes memory) {
     ┆  ┆
 124 │  │                 if (returndata.length == 0 && target.code.length
== 0) {
     │  │                             ────────────┬────────────
     │  │                                         ╰────────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆  ┆
 127 │  │                 return returndata;
     │  │                 ────────────┬────────────
     │  │                             ╰────────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆  ┆
 129 │  ├────▶      }
     │  │
     │  ╰────────────────────────────── Function call
───────╯
Bounds: Bounds for function: function functionCallWithValue(address, bytes,
uint256)
Bounds: Bounds for subcontext: functionCallWithValue(address, bytes,
uint256).fork{ false }.verifyCallResultFromTarget(address, bool,
bytes).fork{ false }.fork{ false }.resume{ functionCallWithValue(address,
bytes, uint256) } where:
1. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
2. (tmp_uint256 >= value) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:83:118]
     │
  83 │    ╭──▶      function functionCallWithValue(address target, bytes
memory data, uint256 value) internal returns (bytes memory) {
     ┆    ┆
  88 │    │             return verifyCallResultFromTarget(target, success,
returndata);
     │    │
─────────────────────────────────────────────────────┬─────────────────────────────────────────
─────
     │    │
 ╰──────────────────────────────────────────────────────────────────── returns:
"verifyCallResultFromTarget(address, bool, bytes).0" ∈ [ {len: 0, indices:
{}}, {len: 2**256 - 1, indices: {}} ]
  89 │    ├──▶       }
     │    │
     │    ╰────────────────────── Entry function call
     │
 118 │  ╭─────▶        ) internal view returns (bytes memory) {
     ┆  ┆
 124 │  │                 if (returndata.length == 0 && target.code.length
== 0) {
     │  │                         ──────────┬──────────
     │  │                                   ╰────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆  ┆
 127 │  │                 return returndata;
     │  │                     ──────────┬──────────
     │  │                               ╰───────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆  ┆
 129 │  ├────▶        }
     │  │
     │  ╰───────────────────────────── Function call
─────╯

Bounds: Bounds for function: function functionStaticCall(address, bytes)
Bounds: Bounds for subcontext: functionStaticCall(address,
bytes).verifyCallResultFromTarget(address, bool, bytes).fork{ false }.fork{
false }.resume{ functionStaticCall(address, bytes) } where:
1. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None
```

```
 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:95:105]
     │
  95 │    ╭─▶     function functionStaticCall(address target, bytes memory
data) internal view returns (bytes memory) {
     ┆    ┆
  97 │    │          return verifyCallResultFromTarget(target, success,
returndata);
     │    │
───────────────────────────────────────────────────────────────────────────
─────
     │    │
 ╰─────────────────────────────────────────────────────────────── returns:
"verifyCallResultFromTarget(address, bool, bytes).0" ∈ [ {len: 0, indices:
{}}, {len: 2**256 - 1, indices: {}} ]
  98 │    ├─▶     }
     │    │
     │    ╰───────────────────── Entry function call
     │
 118 │  ╭──────▶     ) internal view returns (bytes memory) {
     ┆  ┆
 124 │  │                if (returndata.length == 0 && target.code.length
== 0) {
     │  │                                ──────────┬──────────
     │  │                                          ╰────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆  ┆
 127 │  │                return returndata;
     │  │                ──────────┬──────────
     │  │                          ╰────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆  ┆
 129 │  ├────▶     }
     │  │
     │  ╰──────────────────────────── Function call
─────╯
Bounds: Bounds for function: function functionDelegateCall(address, bytes)
Bounds: Bounds for subcontext: functionDelegateCall(address,
bytes).verifyCallResultFromTarget(address, bool, bytes).fork{ false }.fork{
false }.resume{ functionDelegateCall(address, bytes) } where:
1. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None
```

```
╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:104:102]
     │
 104 │    ╭─▶     function functionDelegateCall(address target, bytes
memory data) internal returns (bytes memory) {
     ┆    ┆
 106 │    │          return verifyCallResultFromTarget(target, success,
returndata);
     │    │
────────────────────────────────────────┬─────────────────────────────────
─────
     │    │
╰──────────────────────────────────────────────────────────── returns:
"verifyCallResultFromTarget(address, bool, bytes).0" ∈ [ {len: 0, indices:
{}}, {len: 2**256 - 1, indices: {}} ]
 107 │    ├─▶     }
     │    │
     │    ╰───────────────────── Entry function call
     │
 118 │  ╭─────▶      ) internal view returns (bytes memory) {
     ┆  ┆
 124 │  │                 if (returndata.length == 0 && target.code.length
== 0) {
     │  │                     ──────────┬──────────
     │  │                               ╰────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆  ┆
 127 │  │                 return returndata;
     │  │                 ──────────┬──────────
     │  │                           ╰──────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆  ┆
 129 │  ├─────▶      }
     │  │
     │  ╰──────────────────────────────── Function call
─────╯
Bounds: Bounds for function: function verifyCallResultFromTarget(address,
bool, bytes)
Bounds: Bounds for subcontext: verifyCallResultFromTarget(address, bool,
bytes).fork{ false }.fork{ false } where:
1. (!success == false) == true
```

```
2. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:118:44]
     │
 118 │ ╭─▶     ) internal view returns (bytes memory) {
     ┆ ┆
 127 │ │               return returndata;
     │ │               ──────────┬──────────
     │ │                         ╰────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆ ┆
 129 │ ├─▶     }
     │ │
     │ ╰──────────────────────── Entry function call
─────╯
Bounds: Bounds for function: function verifyCallResult(bool, bytes)
Bounds: Bounds for subcontext: verifyCallResult(bool, bytes).fork{ false }
where:
1. (!success == false) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:135:107]
     │
 135 │ ╭─▶     function verifyCallResult(bool success, bytes memory
returndata) internal pure returns (bytes memory) {
     ┆ ┆
 139 │ │               return returndata;
     │ │               ──────────┬──────────
     │ │                         ╰────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆ ┆
 141 │ ├─▶     }
     │ │
     │ ╰──────────────────────── Entry function call
─────╯
Bounds: Bounds for function: function _revert(bytes)

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:146:60]
```

```
     │
 156 │             revert FailedInnerCall();
     │             ──────────────┬──────────────
     │                           ╰──────────────────────── Execution
guaranteed to revert here!
───────╯
Bounds: Bounds for function: function safeTransfer(IERC20, address,
uint256)
Bounds: Bounds for subcontext: safeTransfer(IERC20, address,
uint256)._callOptionalReturn(IERC20, bytes).functionCall(address,
bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
safeTransfer(IERC20, address, uint256) } where:
1. (returndata.length != 0 && !tmp_bool == false) == true
2. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
36:77]
     │
  36 │    ╭─▶      function safeTransfer(IERC20 token, address to, uint256
value) internal {
     ┆    ┆
  38 │    ├─▶      }
     │    │
     │    ╰───────────────────── Entry function call
     │
  91 │  ╭────▶      function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ┆  ┆
 100 │  ├────▶      }
     │  │
     │  ╰──────────────────────── Function call
     │

 ├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
Address.sol:124:17]
     │
 118 │  ╭──▶      ) internal view returns (bytes memory) {
     ┆ ┆
 124 │ │                if (returndata.length == 0 && target.code.length ==
0) {
     │ │                    ──────────────┬──────────────
     │ │                                  ╰─────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆ ┆
 127 │ │                return returndata;
     │ │                ──────────────┬──────────────
     │ │                              ╰─────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆ ┆
 129 │ ├─▶      }
     │ │
     │ ╰───────────────────────── Function call
 ────╯

Bounds: Bounds for function: function safeTransferFrom(IERC20, address,
address, uint256)
Bounds: Bounds for subcontext: safeTransferFrom(IERC20, address, address,
uint256)._callOptionalReturn(IERC20, bytes).functionCall(address,
bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
safeTransferFrom(IERC20, address, address, uint256) } where:
1. (returndata.length != 0 && !tmp_bool == false) == true
2. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
44:95]
     │
  44 │    ╭─▶    function safeTransferFrom(IERC20 token, address from,
address to, uint256 value) internal {
     ┆    ┆
  46 │    ├─▶    }
     │    │
     │    ╰───────────────────────── Entry function call
     │
  91 │ ╭──────▶    function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ¦  ¦
 100 │  ├───▶     }
     │  │
     │  ╰──────────────────────── Function call
     │

├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
Address.sol:124:17]
     │
 118 │  ╭──▶      ) internal view returns (bytes memory) {
     ¦  ¦
 124 │  │              if (returndata.length == 0 && target.code.length ==
0) {
     │  │                 ──────────┬──────────
     │  │                           ╰─────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ¦  ¦
 127 │  │              return returndata;
     │  │              ──────────┬──────────
     │  │                        ╰─────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ¦  ¦
 129 │  ├─▶      }
     │  │
     │  ╰──────────────────────── Function call
──────╯
Bounds: Bounds for function: function safeIncreaseAllowance(IERC20,
address, uint256)
Bounds: Bounds for subcontext: safeIncreaseAllowance(IERC20, address,
uint256).allowance(address, address).resume{ safeIncreaseAllowance(IERC20,
address, uint256) }.forceApprove(IERC20, address, uint256).fork{ true
}._callOptionalReturnBool(IERC20, bytes).resume{ forceApprove(IERC20,
address, uint256) }._callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
forceApprove(IERC20, address, uint256) }._callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
```

```
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
forceApprove(IERC20, address, uint256) }.resume{
safeIncreaseAllowance(IERC20, address, uint256) } where:
1. (_callOptionalReturnBool(IERC20, bytes).0 == false) == true
2. (oldAllowance <=
(115792089237316195423570985008687907853269984665640564039457584007913129639935 - value)) == true
3. (oldAllowance <= 2**256 - 1 - value) == true
4. (returndata.length != 0 && !tmp_bool == false) == true
5. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
52:91]
     │
  52 │    ╭─▶     function safeIncreaseAllowance(IERC20 token, address
spender, uint256 value) internal {
     ┆    ┆
  55 │    ├─▶     }
     │    │
     │    ╰──────────────────────── Entry function call
     │
  91 │  ╭──────▶     function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ┆  ┆
  96 │  │             bytes memory returndata =
address(token).functionCall(data);
     │  │
─────────────────────────────────────┬──────────────────────────────────────
─
     │  │
 ╰─────────────────────────────────────────────────────────── Memory var
"returndata" == {len: 0, indices: {}}
     ┆  ┆
 100 │  ├────▶     }
     │  │
     │  ╰──────────────────────────── Function call
     │

 ├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
Address.sol:124:17]
```

```
     │
  83 │    ╭─▶     function functionCallWithValue(address target, bytes
memory data, uint256 value) internal returns (bytes memory) {
     ┆    ┆
  89 │    ├─▶     }
     │    │
     │    ╰─────────────────────── Function call
     │
 118 │  ╭────▶     ) internal view returns (bytes memory) {
     ┆  ┆
 124 │  │                  if (returndata.length == 0 && target.code.length
== 0) {
     │  │                  ──────────┬───────────
     │  │                            ╰───────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     │  │                            │
     │  │                            ╰───────────────────── Memory var
"returndata" == {len: 0, indices: {}}
     ┆  ┆
 127 │  │                  return returndata;
     │  │                  ──────────┬───────────
     │  │                            ╰───────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     │  │                            │
     │  │                            ╰───────────────────── returns: "returndata"
== {len: 0, indices: {}}
     ┆  ┆
 129 │  ├────▶     }
     │  │
     │  ╰─────────────────────────── Function call
─────╯
Bounds: Bounds for subcontext: safeIncreaseAllowance(IERC20, address,
uint256).allowance(address, address).resume{ safeIncreaseAllowance(IERC20,
address, uint256) }.forceApprove(IERC20, address, uint256).fork{ false
}._callOptionalReturnBool(IERC20, bytes).resume{ forceApprove(IERC20,
address, uint256) }.resume{ safeIncreaseAllowance(IERC20, address, uint256)
} where:
1. (!_callOptionalReturnBool(IERC20, bytes).0 == false) == true
2. (oldAllowance <=
(115792089237316195423570985008687907853269984665640564039457584007913129639935 - value)) == true
3. (oldAllowance <= 2**256 - 1 - value) == true
```

```
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
52:91]
    │
 52 │ ╭─▶     function safeIncreaseAllowance(IERC20 token, address
spender, uint256 value) internal {
    ┆ ┆
 55 │ ├─▶     }
    │ │
    │ ╰──────────────────── Entry function call
────╯
Bounds: Bounds for function: function safeDecreaseAllowance(IERC20,
address, uint256)
Bounds: Bounds for subcontext: safeDecreaseAllowance(IERC20, address,
uint256).allowance(address, address).resume{ safeDecreaseAllowance(IERC20,
address, uint256) }.fork{ false }.forceApprove(IERC20, address,
uint256).fork{ true }._callOptionalReturnBool(IERC20, bytes).resume{
forceApprove(IERC20, address, uint256) }._callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
forceApprove(IERC20, address, uint256) }._callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
forceApprove(IERC20, address, uint256) }.resume{
safeDecreaseAllowance(IERC20, address, uint256) } where:
1. (_callOptionalReturnBool(IERC20, bytes).0 == false) == true
2. (currentAllowance >= requestedDecrease) == true
3. (returndata.length != 0 && !tmp_bool == false) == true
4. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
61:103]
    │
 61 │ ╭─▶     function safeDecreaseAllowance(IERC20 token, address
spender, uint256 requestedDecrease) internal {
    ┆    ┆
 69 │    ├─▶     }
    │    │
    │    ╰───────────────────── Entry function call
    │
 91 │  ╭─────▶      function _callOptionalReturn(IERC20 token, bytes memory
data) private {
    ┆  ┆
 96 │  │             bytes memory returndata =
address(token).functionCall(data);
    │  │
──────────────────────────────────────┬──────────────────────────────────────
─
    │  │
╰─────────────────────────────────────────────────────────── Memory var
"returndata" == {len: 0, indices: {}}
    ┆  ┆
100 │  ├────▶     }
    │  │
    │  ╰────────────────────────── Function call
    │
    │

 ├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
Address.sol:124:17]
    │
 83 │    ╭──▶     function functionCallWithValue(address target, bytes
memory data, uint256 value) internal returns (bytes memory) {
    ┆    ┆
 89 │    ├─▶     }
    │    │
    │    ╰───────────────────── Function call
    │
 118 │  ╭─────▶      ) internal view returns (bytes memory) {
    ┆  ┆
 124 │  │                 if (returndata.length == 0 && target.code.length
== 0) {
    │  │                             ──────────┬───────────
    │  │                                       ╰──────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
    │  │                                       │
    │  │                                       ╰──────────────────── Memory var
```

```
"returndata" == {len: 0, indices: {}}
      ┆ ┆
 127  │ │                  return returndata;
      │ │                  ──────────────────────
      │ │                          ╰──────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
      │ │                          │
      │ │                          ╰──────────────────── returns: "returndata"
== {len: 0, indices: {}}
      ┆ ┆
 129  │ ├──────▶      }
      │ │
      │ ╰────────────────────────── Function call
──────╯

Bounds: Bounds for subcontext: safeDecreaseAllowance(IERC20, address,
uint256).allowance(address, address).resume{ safeDecreaseAllowance(IERC20,
address, uint256) }.fork{ false }.forceApprove(IERC20, address,
uint256).fork{ false }._callOptionalReturnBool(IERC20, bytes).resume{
forceApprove(IERC20, address, uint256) }.resume{
safeDecreaseAllowance(IERC20, address, uint256) } where:
1. (!_callOptionalReturnBool(IERC20, bytes).0 == false) == true
2. (currentAllowance >= requestedDecrease) == true
, killed: None

 ╭──[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
61:103]
     │
 61  │ ╭──▶     function safeDecreaseAllowance(IERC20 token, address
spender, uint256 requestedDecrease) internal {
     ┆ ┆
 69  │ ├──▶     }
     │ │
     │ ╰────────────────────── Entry function call
─────╯

Bounds: Bounds for function: function forceApprove(IERC20, address,
uint256)
Bounds: Bounds for subcontext: forceApprove(IERC20, address, uint256).fork{
false }._callOptionalReturnBool(IERC20, bytes).resume{ forceApprove(IERC20,
address, uint256) } where:
1. (!_callOptionalReturnBool(IERC20, bytes).0 == false) == true
, killed: None
```

```
╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
76:82]
     │
 76  │  ╭─▶     function forceApprove(IERC20 token, address spender, uint256
value) internal {
     ┆  ┆
 83  │  ├─▶     }
     │  │
     │  ╰──────────────────────── Entry function call
     │
─────╯
Bounds: Bounds for subcontext: forceApprove(IERC20, address, uint256).fork{
true }._callOptionalReturnBool(IERC20, bytes).resume{ forceApprove(IERC20,
address, uint256) }._callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
forceApprove(IERC20, address, uint256) }._callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
forceApprove(IERC20, address, uint256) } where:
1. (_callOptionalReturnBool(IERC20, bytes).0 == false) == true
2. (returndata.length != 0 && !tmp_bool == false) == true
3. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None

╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
76:82]
     │
  76 │     ╭─▶     function forceApprove(IERC20 token, address spender,
uint256 value) internal {
     ┆     ┆
  83 │     ├─▶     }
     │     │
     │     ╰──────────────────────── Entry function call
     │
  91 │  ╭─────▶     function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ¦ ¦
  96 │ │              bytes memory returndata =
address(token).functionCall(data);
     │ │
───────────────────────────────────────┬────────────────────────────────────────
─
     │ │
 ╰──────────────────────────────────────────────────────────── Memory var
"returndata" == {len: 0, indices: {}}
     ¦ ¦
 100 │ ├────▶     }
     │ │
     │ ╰─────────────────────────── Function call
     │
     │

├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
Address.sol:124:17]
     │
  83 │   ╭───▶     function functionCallWithValue(address target, bytes
memory data, uint256 value) internal returns (bytes memory) {
     ¦   ¦
  89 │   ├───▶     }
     │   │
     │   ╰───────────────────────── Function call
     │
 118 │ ╭──────▶     ) internal view returns (bytes memory) {
     ¦ ¦
 124 │ │                   if (returndata.length == 0 && target.code.length
== 0) {
     │ │                          ───────────┬───────────
     │ │                                     ╰──────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     │ │                                     │
     │ │                                     ╰──────────────────── Memory var
"returndata" == {len: 0, indices: {}}
     ¦ ¦
 127 │ │                   return returndata;
     │ │                   ──────────┬──────────
     │ │                             ╰──────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     │ │                             │
     │ │                             ╰──────────────────── returns: "returndata"
```

```
== {len: 0, indices: {}}
      ┆  ┆
  129 │  ├───▶      }
      │  │
      │  ╰────────────────────────── Function call
──────╯

Bounds: Bounds for function: function _callOptionalReturn(IERC20, bytes)
Bounds: Bounds for subcontext: _callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false } where:
1. (returndata.length != 0 && !tmp_bool == false) == true
2. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
, killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils
/Address.sol:124:17]
      │
  118 │  ╭─▶      ) internal view returns (bytes memory) {
      ┆  ┆
  124 │  │              if (returndata.length == 0 && target.code.length ==
0) {
      │  │                    ──────────┬──────────
      │  │                              ╰────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
      ┆  ┆
  127 │  │              return returndata;
      │  │              ──────────┬──────────
      │  │                        ╰────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
      ┆  ┆
  129 │  ├─▶      }
      │  │
      │  ╰────────────────────── Function call
      │

 ├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:9
1:75]
      │
   91 │  ╭─▶      function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ┆  ┆
 100 │  ├─▶     }
     │  │
     │  ╰──────────────────── Entry function call
─────╯

Bounds: Bounds for function: function _callOptionalReturnBool(IERC20,
bytes)
Bounds: Bounds for subcontext: _callOptionalReturnBool(IERC20, bytes),
killed: None

 ╭─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:
110:94]
     │
 110 │  ╭─▶     function _callOptionalReturnBool(IERC20 token, bytes memory
data) private returns (bool) {
     ┆  ┆
 116 │  │           return success && (returndata.length == 0 ||
abi.decode(returndata, (bool))) && address(token).code.length > 0;
     │  │
──────────────────────────────────────────────────────────────────────┬──
─────────────────────────────────────────────────────────────────
     │  │
 ╰────────────────────────────────────────────────────────────────────────
──────────────────── returns: "success && returndata.length == 0 ||
tmp_bool && tmp_bytes.length > 0" ∈ [ false, true ]
 117 │  ├─▶     }
     │  │
     │  ╰──────────────────── Entry function call
─────╯

Bounds: Bounds for function: function depositLiquidity(uint256)
Bounds: Bounds for subcontext: depositLiquidity(uint256).fork{ false
}.safeTransferFrom(IERC20, address, address,
uint256)._callOptionalReturn(IERC20, bytes).functionCall(address,
bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
safeTransferFrom(IERC20, address, address, uint256) }.resume{
depositLiquidity(uint256) } where:
1. (_amount != 0) == true
```

```
2. (returndata.length != 0 && !tmp_bool == false) == true
3. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
4. (totalDeposited <=
(115792089237316195423570985008687907853269984665640564039457584007913129639935 - _amount)) == true
5. (totalDeposited <= 2**256 - 1 - _amount) == true
6. (userDeposit[msg.sender] <=
(115792089237316195423570985008687907853269984665640564039457584007913129639935 - _amount)) == true
7. (userDeposit[msg.sender] <= 2**256 - 1 - _amount) == true
, killed: None
      ╭─[src/simple-perpetual-exchange.sol:100:55]
      │
 100  │  ╭─▶     function depositLiquidity(uint256 _amount) public {
      ┆  ┆
 114  │  ├─▶     }
      │  │
      │  ╰─────────────────────── Entry function call
      │

├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/Address.sol:124:17]
      │
 118  │  ╭─▶     ) internal view returns (bytes memory) {
      ┆  ┆
 124  │  │              if (returndata.length == 0 && target.code.length == 0) {
      │  │                   ──────────┬──────────
      │  │                             ╰─────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
      ┆  ┆
 127  │  │              return returndata;
      │  │              ─────────┬────────
      │  │                       ╰─────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
      ┆  ┆
 129  │  ├─▶     }
      │  │
      │  ╰─────────────────────── Function call
      │

├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:9
1:75]
     │
  91 │ ╭──▶     function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ┆ ┆
 100 │ ├──▶     }
     │ │
     │ ╰───────────────────── Function call
     │
─────╯
Bounds: Bounds for function: function withdrawLiquidity(uint256)
Bounds: Bounds for subcontext: withdrawLiquidity(uint256).fork{ false
}.checkLiquidityWithdraw(uint256).fork{ false }.fork{ false }.fork{ false
}.resume{ withdrawLiquidity(uint256) }.safeTransfer(IERC20, address,
uint256)._callOptionalReturn(IERC20, bytes).functionCall(address,
bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
safeTransfer(IERC20, address, uint256) }.resume{ withdrawLiquidity(uint256)
} where:
1. (_amount != 0) == true
2. (liquidityRatioAfter >= MINIMUM_RESERVE_RATIO) == true
3. (netSupply <= (2**256 - 1) / LIQUIDITY_SCALE) == true
4. (netSupply <=
115792089237316195423570985008687907853269984665640564039457584007913129639
935) == true
5. (netSupply == 0 && totalBorrowed_ == 0 == false) == true
6. (netSupply == 0 && totalBorrowed_ > 0 == false) == true
7. (returndata.length != 0 && !tmp_bool == false) == true
8. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
9. (totalBorrowed_ != 0) == true
10. (totalDeposited >= _amount) == true
11. (userDeposit[msg.sender] >= _amount) == true
, killed: None
     ╭─[src/simple-perpetual-exchange.sol:117:56]
     │
 117 │ ╭──▶     function withdrawLiquidity(uint256 _amount) public {
     ┆ ┆
 130 │ ├──▶     }
     │ │
     │ ╰───────────────────── Entry function call
     │
     │
    
├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
Address.sol:124:17]
     │
 118 │   ╭─▶     ) internal view returns (bytes memory) {
     ┆   ┆
 124 │   │             if (returndata.length == 0 && target.code.length ==
0) {
     │   │                 ──────────┬──────────
     │   │                           ╰────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆   ┆
 127 │   │             return returndata;
     │   │             ──────────┬──────────
     │   │                       ╰────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆   ┆
 129 │   ├─▶     }
     │   │
     │   ╰──────────────────────── Function call
     │
    
├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:9
1:75]
     │
  91 │   ╭─▶     function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ┆   ┆
 100 │   ├─▶     }
     │   │
     │   ╰──────────────────────── Function call
─────╯
Bounds: Bounds for function: function getWBTCPrice()
Bounds: Bounds for subcontext: getWBTCPrice().latestRoundData().resume{
getWBTCPrice() }, killed: None
     ╭─[src/simple-perpetual-exchange.sol:133:58]
     │
 133 │   ╭─▶     function getWBTCPrice() public view returns (uint256){
     ┆   ┆
 135 │   │         return uint256(answer);
     │   │         ──────────────┬──────────────
     │ │                      └─────────────────────── returns:
"uint256(answer)" ∈ [ 0, 2**80 - 1 ]
 136 │ ├─→     }
     │ │
     │ └───────────────────── Entry function call
─────┘

Bounds: Bounds for function: function openPosition(bool, uint256, uint256)
Bounds: Bounds for subcontext: openPosition(bool, uint256, uint256).fork{
false }.fork{ false }.fork{ false
}.getWBTCPrice().latestRoundData().resume{ getWBTCPrice() }.resume{
openPosition(bool, uint256, uint256) }.checkLiquidityBorrow(uint256).fork{
false }.resume{ openPosition(bool, uint256, uint256)
}.safeTransferFrom(IERC20, address, address,
uint256)._callOptionalReturn(IERC20, bytes).functionCall(address,
bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
safeTransferFrom(IERC20, address, address, uint256) }.resume{
openPosition(bool, uint256, uint256) } where:
1. (_collateralAmountUsdc <= (2**256 - 1) / _leverage) == true
2. (_collateralAmountUsdc <=
115792089237316195423570985008687907853269984665640564039457584007913129639
935) == true
3. (_collateralAmountUsdc >= MINIMUM_COLLATERAL) == true
4. (_leverage == 0 || _collateralAmountUsdc == 0 == false) == true
5. (borrowAmountUsdc_ <= (2**256 - 1) / WBTC_DIVISION_SCALE) == true
6. (borrowAmountUsdc_ <=
115792089237316195423570985008687907853269984665640564039457584007913129639
935) == true
7. (liquidityRatioAfter >= MINIMUM_RESERVE_RATIO) == true
8. (netBorrow != 0) == true
9. (returndata.length != 0 && !tmp_bool == false) == true
10. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
11. (totalBorrowed <=
(11579208923731619542357098500868790785326998466564056403945758400791312963
9935 - _amount)) == true
12. (totalBorrowed <=
(11579208923731619542357098500868790785326998466564056403945758400791312963
9935 - borrowAmountWbtc_)) == true
13. (totalBorrowed <= 2**256 - 1 - _amount) == true
```

```
14. (totalBorrowed <= 2**256 - 1 - borrowAmountWbtc_) == true
15. (totalDeposited <= (2**256 - 1) / LIQUIDITY_SCALE) == true
16. (totalDeposited <=
115792089237316195423570985008687907853269984665640564039457584007913129639
935) == true
17. (userPositions[msg.sender].borrowAmountUsdc <= 0) == true
18. (wbtcPrice_ != 0) == true
, killed: None
      ╭─[src/simple-perpetual-exchange.sol:143:16]
      │
  143 │   ╭─▶         public {
      ┆   ┆
  181 │   ├─▶     }
      │   │
      │   ╰────────────────────── Entry function call
      │

 ├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
Address.sol:124:17]
      │
  118 │   ╭─▶     ) internal view returns (bytes memory) {
      ┆   ┆
  124 │   │             if (returndata.length == 0 && target.code.length ==
0) {
      │   │                 ──────────┬──────────
      │   │                           ╰────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
      ┆   ┆
  127 │   │             return returndata;
      │   │             ──────────┬──────────
      │   │                       ╰────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
      ┆   ┆
  129 │   ├─▶     }
      │   │
      │   ╰────────────────────── Function call
      │

 ├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:9
1:75]
      │
   91 │   ╭─▶     function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ┆  ┆
 100 │  ├─▸      }
     │  │
     │  ╰────────────────────── Function call
─────╯

Bounds: Bounds for function: function increasePosition(uint256)
Bounds: Bounds for subcontext: increasePosition(uint256).fork{ false
}.fork{ false }.getWBTCPrice().latestRoundData().resume{ getWBTCPrice()
}.resume{ increasePosition(uint256) }.checkLiquidityBorrow(uint256).fork{
false }.resume{ increasePosition(uint256) }.safeTransferFrom(IERC20,
address, address, uint256)._callOptionalReturn(IERC20,
bytes).functionCall(address, bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
safeTransferFrom(IERC20, address, address, uint256) }.resume{
increasePosition(uint256) } where:
1. (_collateralAmountUsdc != 0) == true
2. (_collateralAmountUsdc <= (2**256 - 1) / WBTC_DIVISION_SCALE) == true
3. (_collateralAmountUsdc <= (2**256 - 1) / userPosition_.leverage) == true
4. (_collateralAmountUsdc <=
115792089237316195423570985008687907853269984665640564039457584007913129639
935) == true
5. (_collateralAmountUsdc >= MINIMUM_COLLATERAL) == true
6. (liquidityRatioAfter >= MINIMUM_RESERVE_RATIO) == true
7. (netBorrow != 0) == true
8. (returndata.length != 0 && !tmp_bool == false) == true
9. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
10. (totalBorrowed <=
(11579208923731619542357098500868790785326998466564056403945758400791312963
9935 - _amount)) == true
11. (totalBorrowed <=
(11579208923731619542357098500868790785326998466564056403945758400791312963
9935 - borrowAmountWbtc_)) == true
12. (totalBorrowed <= 2**256 - 1 - _amount) == true
13. (totalBorrowed <= 2**256 - 1 - borrowAmountWbtc_) == true
14. (totalDeposited <= (2**256 - 1) / LIQUIDITY_SCALE) == true
15. (totalDeposited <=
115792089237316195423570985008687907853269984665640564039457584007913129639
935) == true
```

```
16. (userPosition_.borrowAmountUsdc <=
(115792089237316195423570985008687907853269984665640564039457584007913129639935 - borrowAmountUsdc_)) == true
17. (userPosition_.borrowAmountUsdc <= 2**256 - 1 - borrowAmountUsdc_) ==
true
18. (userPosition_.borrowAmountWbtc <=
(115792089237316195423570985008687907853269984665640564039457584007913129639935 - borrowAmountWbtc_)) == true
19. (userPosition_.borrowAmountWbtc <= 2**256 - 1 - borrowAmountWbtc_) ==
true
20. (wbtcPrice_ != 0) == true
, killed: None
     ╭─[src/simple-perpetual-exchange.sol:184:69]
     │
 184 │ ╭─▶     function increasePosition(uint256 _collateralAmountUsdc)
public {
     ┆ ┆
 214 │ ├─▶     }
     │ │
     │ ╰───────────────────── Entry function call
     │

├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
Address.sol:124:17]
     │
 118 │ ╭─▶     ) internal view returns (bytes memory) {
     ┆ ┆
 124 │ │               if (returndata.length == 0 && target.code.length ==
0) {
     │ │                   ──────────┬──────────
     │ │                             ╰─────────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆ ┆
 127 │ │               return returndata;
     │ │               ──────────┬──────────
     │ │                         ╰─────────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆ ┆
 129 │ ├─▶     }
     │ │
     │ ╰───────────────────── Function call
     │
 ├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:9
1:75]
     │
  91 │  ╭─▶     function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ┆  ┆
 100 │  ├─▶     }
     │  │
     │  ╰───────────────────── Function call
─────╯
Bounds: Bounds for function: function increaseCollateral(uint256)
Bounds: Bounds for subcontext: increaseCollateral(uint256).fork{ false
}.safeTransferFrom(IERC20, address, address,
uint256)._callOptionalReturn(IERC20, bytes).functionCall(address,
bytes).functionCallWithValue(address, bytes,
uint256).verifyCallResultFromTarget(address, bool, bytes).fork{ false
}.fork{ false }.resume{ functionCallWithValue(address, bytes, uint256)
}.resume{ functionCall(address, bytes) }.resume{
_callOptionalReturn(IERC20, bytes) }.fork{ false }.resume{
safeTransferFrom(IERC20, address, address, uint256) }.resume{
increaseCollateral(uint256) } where:
1. (_collateralAmountUsdc != 0) == true
2. (returndata.length != 0 && !tmp_bool == false) == true
3. (returndata.length == 0 && tmp_bytes.length == 0 == false) == true
4. (userPositions[msg.sender].collateralAmount <=
(1157920892373161954235709850086879078532699846656405640394575840079131296
9935 - _collateralAmountUsdc)) == true
5. (userPositions[msg.sender].collateralAmount <= 2**256 - 1 -
_collateralAmountUsdc) == true
, killed: None
     ╭─[src/simple-perpetual-exchange.sol:217:71]
     │
 217 │  ╭─▶     function increaseCollateral(uint256 _collateralAmountUsdc)
public {
     ┆  ┆
 226 │  ├─▶     }
     │  │
     │  ╰───────────────────── Entry function call
     │

 ├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/../../../utils/
```

```
Address.sol:124:17]
     │
 118 │ ╭──▶     ) internal view returns (bytes memory) {
     ┆ ┆
 124 │ │             if (returndata.length == 0 && target.code.length ==
0) {
     │ │                ──────────────┬──────────────
     │ │                              ╰─────────────── Memory var
"returndata" ∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆ ┆
 127 │ │             return returndata;
     │ │             ──────────┬──────────
     │ │                       ╰─────────────── returns: "returndata"
∈ [ {len: 0, indices: {}}, {len: 2**256 - 1, indices: {}} ]
     ┆ ┆
 129 │ ├──▶     }
     │ │
     │ ╰───────────────────────── Function call
     │

├─[./lib/openzeppelin-contracts/contracts/token/ERC20/utils/SafeERC20.sol:9
1:75]
     │
  91 │ ╭──▶     function _callOptionalReturn(IERC20 token, bytes memory
data) private {
     ┆ ┆
 100 │ ├──▶     }
     │ │
     │ ╰───────────────────────── Function call
─────╯
Bounds: Bounds for function: function checkLiquidityBorrow(uint256)
Bounds: Bounds for subcontext: checkLiquidityBorrow(uint256).fork{ false }
where:
1. (liquidityRatioAfter >= MINIMUM_RESERVE_RATIO) == true
2. (netBorrow != 0) == true
3. (totalBorrowed <=
(1157920892373161954235709850086879078532699846656405640394575840079131296
39935 - _amount)) == true
4. (totalBorrowed <= 2**256 - 1 - _amount) == true
5. (totalDeposited <= (2**256 - 1) / LIQUIDITY_SCALE) == true
6. (totalDeposited <=
115792089237316195423570985008687907853269984665640564039457584007913129639
```

```
935) == true
, killed: None
     ╭─[src/simple-perpetual-exchange.sol:229:64]
     │
 229 │ ╭─▶     function checkLiquidityBorrow(uint256 _amount) public view
{
     ┆ ┆
 235 │ ├─▶     }
     │ │
     │ ╰──────────────────── Entry function call
─────╯
Bounds: Bounds for function: function checkLiquidityWithdraw(uint256)
Bounds: Bounds for subcontext: checkLiquidityWithdraw(uint256).fork{ false
}.fork{ false }.fork{ false } where:
1. (liquidityRatioAfter >= MINIMUM_RESERVE_RATIO) == true
2. (netSupply <= (2**256 - 1) / LIQUIDITY_SCALE) == true
3. (netSupply <=
115792089237316195423570985008687907853269984665640564039457584007913129639
935) == true
4. (netSupply == 0 && totalBorrowed_ == 0 == false) == true
5. (netSupply == 0 && totalBorrowed_ > 0 == false) == true
6. (totalBorrowed_ != 0) == true
7. (totalDeposited >= _amount) == true
, killed: None
     ╭─[src/simple-perpetual-exchange.sol:238:66]
     │
 238 │ ╭─▶     function checkLiquidityWithdraw(uint256 _amount) public
view {
     ┆ ┆
 253 │ ├─▶     }
     │ │
     │ ╰──────────────────── Entry function call
     │
```