ADVANCED TOPICS IN SOFTWARE ENGINEERING CSE 6324-001

FINAL PRESENTATION

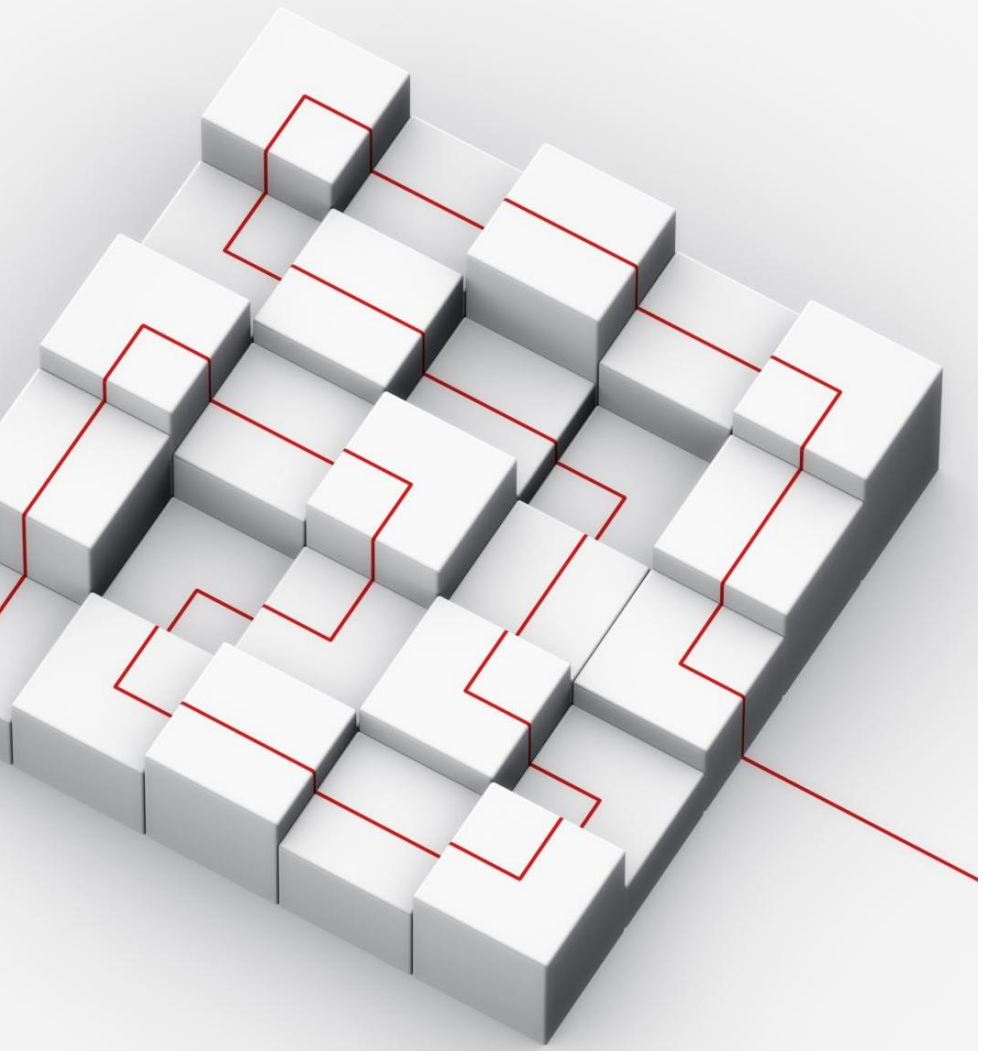
TEAM 9

ABHISHEK WADHWANI – 10020352719

MOUNIKA KOTTAPALLI – 1002085510

NITIN RAJ THUMMA – 1002080555

SAI RAGHU RAMI REDDY DONTIREDDY - 1002014523



PROJECT PROPOSAL

The proposal aims to enhance Slither Framework Analysis by incorporating functionality to handle nested structs in Solidity contracts. By examining Solidity language specifications, we gain insight into the structure and usage of nested structs, allowing us to develop a deep understanding of the logic and patterns associated with them. [1][2]

The goal is to strengthen Slither's effectiveness in evaluating Solidity code by connecting a detection algorithm to its existing architecture. This algorithm efficiently parses through the AST, identifying instances of nested structs in variable declarations and improving Slither's precision.

ISSUE

Slither-read-storage: Add support for Nested Structs #2077

Enhancing Slither with nested struct support would enable broader Solidity contract security analysis.

This improvement will expand Slither's scope and accuracy in evaluating contracts, making it more comprehensive and precise.

It addresses the current limitation in analyzing security vulnerabilities within smart contracts that utilize nested structs. [2]

DETECTOR CODE

```python
@staticmethod
def _find_struct_var_slot(
    elems: List[StructureVariable], slot_as_bytes: bytes, struct_var: str
) -> Tuple[str, str, bytes, int, int]:
    """
    Finds the slot of a structure variable.
    Args:
        elems (List[StructureVariable]): Ordered list of structure variables.
        slot_as_bytes (bytes): The slot of the struct to begin searching at.
        struct_var (str): The target structure variable.
    Returns:
        info (str): Info about the target variable to log.
        type_to (str): The type of the target variable.
        slot (bytes): The storage location of the target variable.
        size (int): The size (in bits) of the target variable.
        offset (int): The size of other variables that share the same slot.
    """
    slot = int.from_bytes(slot_as_bytes, "big")
    offset = 0
    type_to = ""
    size = 0
    for var in elems:
        var_type = var.type
        if isinstance(var_type, list):
            nested_info, nested_type_to, nested_slot_as_bytes, nested_size, nested_offset = _find_struct_var_slot(
                var_type, slot_as_bytes, struct_var
            )
            if nested_info:
                return nested_info, nested_type_to, nested_slot_as_bytes, nested_size, nested_offset
        else:
            size = var_type.size
            if offset >= 256:
                slot += 1
                offset = 0
            if struct_var == var.name:
                type_to = var_type
                break
            offset += size

    slot_as_bytes = int.to_bytes(slot, 32, byteorder="big")
    info = f"\nStruct Variable: {struct_var}"
    return info, type_to, slot_as_bytes, size, offset
```

ABOUT THE DETECTOR

1. Parameters:

elems: List of StructureVariable objects representing ordered structure variables.
slot_as_bytes: Bytes representing the initial storage slot to start the search.
struct_var: The target structure variable to search for within the list.

2. Return Values:

info: Information about the target variable to log.
type_to: The type of the target variable.
slot: Bytes representing the storage location of the target variable.
size: The size (in bits) of the target variable.
offset: The size of other variables that share the same slot.

SPECIFICATION AND DESIGN

- Tools such as Hardhat, Python, Pip3, and Slither, establishing a solid foundation were installed for this project[3].
- We deployed a Smart Contract on the Polygon Mumbai blockchain and worked to enhance Slither's functionality for analyzing contracts that incorporate nested structs. [6]

A. Figuring out blockchains onto which we can deploy on smart contract

After considering many options we chose Polygon Mumbai because it is scalable, efficient and developer friendly.

B. Nested Struct Smart Contracts

There are 2 structs in it which are home address and person.

C. Installing and Creating a dotenv file

- dotenv is used to store sensitive or configuration-related information separately from the code by using environment variables. These variables are then accessed within the code to provide security and flexibility. [9]

```
2   PRIVATE_KEY=your private key
3   ACCOUNT_ADDRESS=your account address
4   ETHERSCAN_API_FOR_POLYGON=your etherscan api key
```

```
1   PRIVATE_KEY=02b98c30c7697190b3cc9c50f0569cea9e7ea21433781595fc739c9bf417650c
2   ACCOUNT_ADDRESS=0xf9F6febD167C37E828C8B276b5aD8EC245Ffe987
```

D. Writing Deployment Scripts for Hardhat Tool

i. Install the Hardhat toolbox. [3]
ii. Import the dotenv package. [9]
iii. Load the environment variables from the .env file into the current process.

iv. Specify the RPC URL of the Mumbai testnet. [6]
v. Specify the chainID Field (Chain ID) of Mumabi testnet. [6]
vi. Specify the private keys of the account that Hardhat should use to deploy and test your contracts on the Mumbai testnet. [7]

E. Command for deploying the contract on the polygon blockchain

The smart contract is deployed on polygon mumbai testnet with an address of 0xB091F9c43b4D58C6040F65841E4DF8Be27726BC2 [9]
https://mumbai.polygonscan.com/address/0xB091F9c43b4D58C6040F65841E4DF8Be27726BC2

G Input and Output for the detector

The contract provides two functions: addPerson() and getPerson().

The addPerson() function allows users to add a new person to the people mapping.

- The getPerson() function allows users to get the information of the person who is calling the function.

NestedStructs.sol

Slither_github_issue > contracts > NestedStructs.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.18;

contract NestedStructExample {
    struct HomeAddress {
        string street;
        string city;
        string country;
    }

    struct Person {
        string name;
        uint age;
        HomeAddress location;
    }

    mapping(address => Person) public people;

    function addPerson(string memory name, uint age, string memory street, string memory city, string memory country) public {
        HomeAddress memory addr = HomeAddress(street, city, country);
        Person memory newPerson = Person(name, age, addr);
        people[msg.sender] = newPerson;
    }

    function getPerson() public view returns (string memory, uint, string memory, string memory, string memory) {
        Person storage person = people[msg.sender];
        return (person.name, person.age, person.location.street, person.location.city, person.location.country);
    }
}
```

H. Running the command for the Detector on the Deployed Contract

- Need to mention mumbai and then the address as the contract is deployed on Polygon Mumbai Testnet. [8]

```
INFO:Slither-read-storage:
Contract 'NestedStructExample'
NestedStructExample.people with type mapping(address => NestedStructExample.Person) is located at slot: 0

INFO:Slither-read-storage:
Name: people
Type: mapping(address => NestedStructExample.Person)
Slot: 0
```

- This isn’t throwing any error as of now for nested structs. It shows slot 0 of our contract as can be seen. We are trying to modify the detector so that it gives further details about how the slots for Person and HomeAddress could be managed.
- The current slither detector is unable to detect the nested Structs.

Error Encountered

```
INFO:Slither-read-storage:<class 'slither.core.solidity_types.user_defined_type.UserDefinedType'> is current not implemented in find_struct_var_slot
Traceback (most recent call last):
  File "/home/abhish/.local/bin/slither-read-storage", line 8, in <module>
    sys.exit(main())
  File "/home/abhish/.local/lib/python3.10/site-packages/slither/tools/read_storage/__main.py", line 157, in main
    srs.get_storage_layout()
  File "/home/abhish/.local/lib/python3.10/site-packages/slither/tools/read_storage/read_storage.py", line 150, in get_storage_layout
    tmp[var.name].elems = self.all_struct_slots(var, type type, contract)
  File "/home/abhish/.local/lib/python3.10/site-packages/slither/tools/read_storage/read_storage.py", line 921, in all_struct_slots
    info = self.get_storage_slot(
  File "/home/abhish/.local/lib/python3.10/site-packages/slither/tools/read_storage/read_storage.py", line 256, in get_storage_slot
    info, type to, slot, size, offset = self.find_struct_var_slot(elems, slot, struct_var)
  File "/home/abhish/.local/lib/python3.10/site-packages/slither/tools/read_storage/read_storage.py", line 601, in find_struct_var_slot
    return info, type to, slot as bytes, size, offset
UnboundLocalError: local variable 'size' referenced before assignment
```

This error means that *slither-read-storage* program is not able to find a slot for the struct variable. This is because the *slither.core.solidity_types.user_defined_type.UserDefinedType* class is not currently implemented in the *find_struct_var_slot* function.
We managed to get rid of the above error by adding changes to the read_storage.py file.

Project Plan

#	Task Description	Status
1.	Installation a) Python v3.6+ b) Hardhat c) Ubuntu OS d) Slither	Completed
2.	Decide the detector to start working on	Completed
3.	a) Find the sample solidity contract to analyze it using Slither b) Work on reviews from Inception	Completed
4.	a) Integrate Slither and Smart Contract with Hardhat. b) Codebase and Folder Structure Analysis.	Completed
5.	a) Load the solidity smart contract. b) Set up the solidity compiler version based on the sample solidity smart contract. c) Work on reviews from Iteration 1	Completed
6.	a) Create a .sol solidity file. b) Installing dotenv c) Create dotenv file d) Contract deployment	Completed
7.	a) Write a detector to detect vulnerability. b) Analyze the solidity contract with nested structs.	Completed
8.	a) Development of the detector b) Work on review from Iteration 2	**C**ompleted
9.	a) Improvising the detector and fixing errors b) Testing and analysis c) Generate reports and documentation for Final Iteration	Incomplete Incomplete Completed

RISK FACTORS AND MITIGATION PLAN

Risk Factor	Mitigation Plan	Risk Exposure
Maintaining and compatibility	Sticking with a fixed version of solidity in the detector.	Trying to maintain the same version and implement the code trying to make it compatible.
Thorough Testing and Validation	Create a detailed list of tests for the nested structs.	Incomplete.
Technical issue - Inexperience with Python	Learning python via tutorials.	With minimal knowledge of programming Language, it is difficult to work on the code.
Technical issue Inexperience with Solidity	Learning solidity language concepts by tutorials.	With no knowledge of Solidity, it is very difficult to work on Nested Struct and it takes time to get hand of it.
Technical issue Complexity Building the Nested Struct and using in a Function	Finding out how to build nested structs and write them in the smart contracts	Analyzing existing solidity open-source code and going through Solidity Documentation and relevant work.
Not receiving the anticipated output	Rechecking and writing the code and find where it went wrong	Analyzing the complete code and trying to enhance the code to get a better outcome.

CUSTOMERS AND USERS

CSE 6324-001 class

Aditya Vichare (CSE 6324-001) - Project is interesting but incomplete due to errors, requiring further refinement.

Smart Contract Enthusiast (Amosh Sapkota) - This concept is presented in a way that is easily grasped by beginners.

GitHub Repository: ***https://github.com/Abhismoothie/Slither-Enhancementproject-team-9-CSE6324-001***

REFERENCES

1. https://github.com/crytic/slither
2. https://github.com/crytic/slither/issues/2077
3. hardhat.org/hardhat-runner/docs/getting-started#overview
4. https://github.com/crytic/slither#how-to-install
5. https://www.immunebytes.com/blog/slither-a-solidity-static-analyzer-for-smart-contracts/
6. https://www.alchemy.com/overviews/mumbai-testnet
7. https://hardhat.org/hardhat-runner/docs/guides/deploying
8. https://docs.chainstack.com/reference/polygon-getting-started
9. https://www.npmjs.com/package/dotenv
10. https://onboardbase.com/blog/env-file-guide/
11. https://docs.arbitrum.io/stylus/how-tos/local-stylus-dev-node
12. https://stackoverflow.com/questions/70011797/return-nested-struct-inside-struct-in-solidity
13. https://github.com/dokzai/slither/commit/f42bbad5f8cc55584a42f514141265f95fecde98

Questions?

Thank You!