# Smart Contract Security Audit Report

Report generated on: 2025-05-21 15:37:09

## Contract Source Code

```
File: dosv.sol

contract VulnerableAirdrop {
    address[] public recipients;
    mapping(address => bool) public hasReceived;

    function addRecipients(address[] memory _recipients) public {
        for(uint i = 0; i < _recipients.length; i++) {
            recipients.push(_recipients[i]);
        }
    }

    function distribute() public {
        for(uint i = 0; i < recipients.length; i++) {
            if(!hasReceived[recipients[i]]) {
                payable(recipients[i]).transfer(1 ether);
                hasReceived[recipients[i]] = true;
            }
        }
    }
}
```

## Executive Summary

The analysis identified 7 potential security issues:

**1. Medium Severity:**

Warning: SPDX licen

se identifier not provided in source file. Befo re publishing, consider adding a comment contai ning "SPDX-License-Identifier: <SPDX-License>"

to each source file. Use "SPDX-License-Identifi er: UNLICENSED" for non-open-source code. Pleas e see https://spdx.org for more information.

**2. Medium Severity:**

W

# Smart Contract Security Audit Report

Report generated on: 2025-05-21 15:37:09

arning: Source file does not specify required c ompiler version! Consider adding "pragma solidi

t
y

^
0
.
8
.
6
;
"

**3. Medium Severity:**

**4. Medium Severity:**

**5. Medium Severity:**

**6. Medium Severity:**

**7. Medium Severity:**

## Detailed Analysis: Slither

'solc --version' running

'solc /src/input.sol --combined-json abi,ast,bin,bin-runtime,srcmap,srcmap-runtime,userdoc,devdoc,hashes,compact-format --allow-paths .,/src' running

Compilation warnings/errors on /src/input.sol:

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: UNLICENSED" for non-open-source code. Please see https://spdx.org for more information.

--> /src/input.sol

Warning: Source file does not specify required compiler version! Consider adding "pragma solidity ^0.8.6;"

--> /src/input.sol

**Security Finding:**

VulnerableAirdrop.distribute() (src/input.sol#11-18) sends eth to arbitrary user

Dangerous calls:

- address(recipients[i]).transfer(1000000000000000000) (src/input.sol#14)

*Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations*

**Security Finding:**

VulnerableAirdrop.distribute() (src/input.sol#11-18) has external calls inside a loop: address(recipients[i]).transfer(1000000000000000000) (src/input.sol#14)

*Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop*

**Security Finding:**

Parameter VulnerableAirdrop.addRecipients(address[])._recipients (src/input.sol#5) is not in mixedCase

*Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions*

**Security Finding:**

Reentrancy in VulnerableAirdrop.distribute() (src/input.sol#11-18):

External calls:

- address(recipients[i]).transfer(1000000000000000000) (src/input.sol#14)

State variables written after the call(s):

- hasReceived[recipients[i]] = true (src/input.sol#15)

*Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4*

**Security Finding:**

Loop condition i < recipients.length (src/input.sol#12) should use cached array length instead of referencing `length` member of the storage array.

*Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cache-array-length*

INFO:Slither:/src/input.sol analyzed (1 contracts with 100 detectors), 5 result(s) found

## Detailed Analysis: Oyente

WARNING:root:You are using evm version 1.8.2. The supported version is 1.7.3

WARNING:root:You are using solc version 0.4.21, The latest supported version is 0.4.19

CRITICAL:root:Solidity compilation failed. Please use -ce flag to see the detail.