Aderyn Analysis Report

This report was generated by Aderyn, a static analysis tool built by Cyfrin, a blockchain security company. This report is not a substitute for manual audit or security review. It should not be relied upon for any purpose other than to assist in the identification of potential security vulnerabilities.

Table of Contents

Summary

Files Summary

Key	Value
.sol Files	1
Total nSLOC	109

Files Details

Filepath	nSLOC
src/TrickOrTreat.sol	109
Total	**109**

Issue Summary

Category	No. of Issues
High	2
Low	9

High Issues

H-1: Functions send eth away from contract but performs no checks on any address.

Consider introducing checks for `msg.sender` to ensure the recipient of the money is as intended.

▶ 2 Found Instances

- Found in src/TrickOrTreat.sol Line: 48

  ```
      function trickOrTreat(string memory _treatName) public payable nonReentrant {
  ```

- Found in src/TrickOrTreat.sol Line: 146

  ```
      function withdrawFees() public onlyOwner {
  ```

H-2: Weak Randomness

The use of keccak256 hash functions on predictable values like block.timestamp, block.number, or similar data, including modulo operations on these values, should be avoided for generating randomness, as they are easily predictable and manipulable. The `PREVRANDAO` opcode also should not be used as a source of randomness. Instead, utilize Chainlink VRF for cryptographically secure and provably random values to ensure protocol integrity.

▶ 2 Found Instances

- Found in src/TrickOrTreat.sol Line: 57

  ```
              uint256(keccak256(abi.encodePacked(block.timestamp, msg.sender, nextTokenId, block.prevrandao))) % 1000 + 1;
  ```

Low Issues

L-1: Centralization Risk for trusted owners

Contracts have owners with privileged rights to perform admin tasks and need to be trusted to not perform malicious updates or drain funds.

▶ 5 Found Instances

- Found in src/TrickOrTreat.sol Line: 9

  ```
  contract SpookySwap is ERC721URIStorage, Ownable(msg.sender), ReentrancyGuard {
  ```

- Found in src/TrickOrTreat.sol Line: 37

  ```
      function addTreat(string memory _name, uint256 _rate, string memory _metadataURI) public onlyOwner {
  ```

- Found in src/TrickOrTreat.sol Line: 43

  ```
      function setTreatCost(string memory _treatName, uint256 _cost) public onlyOwner {
  ```

- Found in src/TrickOrTreat.sol Line: 146

  ```
      function withdrawFees() public onlyOwner {
  ```

- Found in src/TrickOrTreat.sol Line: 156

  ```
      function changeOwner(address _newOwner) public onlyOwner {
  ```

L-2: Unsafe ERC20 Operations should not be used

ERC20 functions may not behave as expected. For example: return values are not always meaningful. It is recommended to use OpenZeppelin's SafeERC20 library.

▶ 1 Found Instances

- Found in src/TrickOrTreat.sol Line: 148

  ```
          payable(owner()).transfer(balance);
  ```

L-3: Solidity pragma should be specific, not wide

Consider using a specific version of Solidity in your contracts instead of a wide version. For example, instead of `pragma solidity ^0.8.0;`, use `pragma solidity 0.8.0;`

▶ 1 Found Instances

- Found in src/TrickOrTreat.sol Line: 2

```
pragma solidity ^0.8.24;
```

L-4: `public` functions not used internally could be marked `external`

Instead of marking a function as `public`, consider marking it as `external` if it is not used internally.

▶ 6 Found Instances

- Found in src/TrickOrTreat.sol Line: 43

```
    function setTreatCost(string memory _treatName, uint256 _cost) public
onlyOwner {
```

- Found in src/TrickOrTreat.sol Line: 48

```
    function trickOrTreat(string memory _treatName) public payable
nonReentrant {
```

- Found in src/TrickOrTreat.sol Line: 118

```
    function resolveTrick(uint256 tokenId) public payable nonReentrant {
```

- Found in src/TrickOrTreat.sol Line: 146

```
    function withdrawFees() public onlyOwner {
```

- Found in src/TrickOrTreat.sol Line: 152

```
    function getTreats() public view returns (string[] memory) {
```

- Found in src/TrickOrTreat.sol Line: 156

```
    function changeOwner(address _newOwner) public onlyOwner {
```

L-5: Event is missing `indexed` fields

Index event fields make the field more quickly accessible to off-chain tools that parse events. However, note that each index field costs extra gas during emission, so it's not necessarily best to index the maximum allowed per event (three fields). Each event should use three indexed fields if there are three or more fields, and gas usage is not particularly of concern for the events in question. If there are fewer than three fields, all of the fields should be indexed.

▶ 3 Found Instances

- Found in src/TrickOrTreat.sol Line: 25

```
    event TreatAdded(string name, uint256 cost, string metadataURI);
```

- Found in src/TrickOrTreat.sol Line: 26

```
    event Swapped(address indexed user, string treatName, uint256 tokenId);
```

- Found in src/TrickOrTreat.sol Line: 27

```
    event FeeWithdrawn(address owner, uint256 amount);
```

L-6: Using `ERC721::_mint()` can be dangerous

Using `ERC721::_mint()` can mint ERC721 tokens to addresses which don't support ERC721 tokens. Use `_safeMint()` instead of `_mint()` for ERC721.

▶ 2 Found Instances

- Found in src/TrickOrTreat.sol Line: 81

```
                _mint(address(this), tokenId);
```

- Found in src/TrickOrTreat.sol Line: 110

```
        _mint(recipient, tokenId);
```

L-7: PUSH0 is not supported by all chains

Solc compiler version 0.8.20 switches the default target EVM version to Shanghai, which means that the generated bytecode will include PUSH0 opcodes. Be sure to select the appropriate EVM version in case you intend to deploy on a chain other than mainnet like L2 chains that may not support PUSH0, otherwise deployment of your contracts will fail.

▶ 1 Found Instances

- Found in src/TrickOrTreat.sol Line: 2

  ```
  pragma solidity ^0.8.24;
  ```

L-8: Costly operations inside loops.

Invoking `SSTORE`operations in loops may lead to Out-of-gas errors. Use a local variable to hold the loop computation result.

▶ 1 Found Instances

- Found in src/TrickOrTreat.sol Line: 32

  ```
          for (uint256 i = 0; i < treats.length; i++) {
  ```

L-9: State variable changes but no event is emitted.

State variable changes in this function but no event is emitted.

▶ 2 Found Instances

- Found in src/TrickOrTreat.sol Line: 43

  ```
      function setTreatCost(string memory _treatName, uint256 _cost) public
  onlyOwner {
  ```

- Found in src/TrickOrTreat.sol Line: 118

  ```
      function resolveTrick(uint256 tokenId) public payable nonReentrant {
  ```