expelee

Building the Futuristic Blockchain Ecosystem

# SECURITY AUDIT REPORT

## SquidTokenV2

# TOKEN OVERVIEW

## Risk Findings

| Severity | Found |
|---|---|
| High | 0 |
| Medium | 1 |
| Low | 1 |
| Informational | 1 |

## Centralization Risks

| Owner Privileges | Description |
|---|---|
| Can Owner Set Taxes >25% ? | Not Detected |
| Owner needs to enable trading ? | Not Detected |
| Can Owner Disable Trades ? | Not Detected |
| Can Owner Mint ? | Not Detected |
| Can Owner Blacklist ? | Not Detected |
| Can Owner set Max Wallet amount ? | Not Detected |
| Can Owner Set Max TX amount ? | Not Detected |

# TABLE OF CONTENTS

# OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

| | |
|---|---|
| **Audit Result** | **Passed** |
| **KYC Verification** | **-** |
| **Audit Date** | **08 April 2024** |

# CONTRACT DETAILS

**Token Address: 0xFAfb7581a65A1f554616Bf780fC8a8aCd2Ab8c9b**

**Name: SquidTokenV2**

**Symbol: SQUID**

**Decimals:** 18

**Network: Base Scan**

**Token Type:** ERC-20

**Owner: 0xff00d2D6210a537B517138389C29c8A6bb56DaD7**

**Deployer: 0xff00d2D6210a537B517138389C29c8A6bb56DaD7**

**Token Supply:** 71,430,240

**Checksum: A2032c616934aeb47e6039f76b20d261**

**Testnet:**
**https://testnet.bscscan.com/address/0x98dE47C1a577558356a702a5267633cC133741dD#code**

# AUDIT METHODOLOGY

## Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

## Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch , that lead to scams and rugpulls.

## Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

## Tools

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

# VULNERABILITY CHECKS

| | |
|---|---|
| Design Logic | Passed |
| Compiler warnings | Passed |
| Private user data leaks | Passed |
| Timestamps dependence | Passed |
| Integer overflow and underflow | Passed |
| Race conditions & reentrancy. Cross-function race conditions | Passed |
| Possible delays in data delivery | Passed |
| Oracle calls | Passed |
| Front Running | Passed |
| DoS with Revert | Passed |
| DoS with block gas limit | Passed |
| Methods execution permissions | Passed |
| Economy model | Passed |
| Impact of the exchange rate on the logic | Passed |
| Malicious event log | Passed |
| Scoping and declarations | Passed |
| Uninitialized storage pointers | Passed |
| Arithmetic accuracy | Passed |
| Cross-function race conditions | Passed |
| Safe Zepplin module | Passed |

# RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and acces control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

## High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

## Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

## Low Risk

Issues on this level are minor details and warning that can remain unfixed.

## Informational

Issues on this level are minor details and warning that can remain unfixed.

expelee

# INHERITANCE TREE

# TESTNET VERSION

1- Approve (passed):
https://testnet.bscscan.com/tx/0xfea5aaa7c9cdbdc3b4f88cbcfc9acfd0ad4b6c3c142f7548372d95919451339a

2- Set Delegate (passed):
https://testnet.bscscan.com/tx/0x71cb446be9eabba57b9d3a9c4c09cf46a2bdc83eb07e7b8ca1934a129e454cd7

3- Set Pre-Crime (passed):
https://testnet.bscscan.com/tx/0xa7577bfa4f3ab070a1efdf6955e4106781a96d03ae6cfefff238f67e1d3f33f0

4- Set Msg Inspector (passed):
https://testnet.bscscan.com/tx/0x8868451110c8428311bd42f016fb45a9beefb422d49f5935c195acac0eee6268

# MANUAL REVIEW

## Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categroies:
High
Medium
Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use

| Overall Risk Severity | | | | |
|---|---|---|---|---|
| Impact | HIGH | Medium | High | Critical |
| | MEDIUM | Low | Medium | High |
| | LOW | Note | Low | Medium |
| | | LOW | MEDIUM | HIGH |
| | Likelihood | | | |

# MEDIUM RISK FINDING

## Centralization – Missing Require Check
## Severity: Medium
## subject: Set Delegate/Msg Inspector
## Status: Open

**Overview:**
The owner can set any arbitrary address excluding zero address as this is not recommended because if the owner sets the address to the contract address, then the ETH will not be sent to that address and the transaction will fail and this will lead to a potential honeypot in the contract.

```
function setMsgInspector(address _msgInspector) public virtual
onlyOwner {
   msgInspector = _msgInspector;
emit MsgInspectorSet(_msgInspector);
  }
function setDelegate(address _delegate) external;
}
```

**Suggestion:**
It is recommended that the address should not be able to be set as a contract address.

# LOW RISK FINDING

## Centralization – Missing Events

**Severity: Low**

**subject: Missing Events**

**Status: Open**

**Overview:**

They serve as a mechanism for emitting and recording data onto the blockchain, making it transparent and easily accessible.

```
function setDelegate(address _delegate) external;
}
```

**Suggestion:**

Emit an event for critical changes.

# INFORMATIONAL & OPTIMIZATIONS

## Optimization

**Severity: Optimization**

**subject: Remove unused code.**

**Status: Open**

**Overview:**

Unused variables are allowed in Solidity, and they do. not pose a direct security issue. It is the best practice. though to avoid them

```
 function sendValue(address payable recipient, uint256 amount)
internal {
if (address(this).balance < amount) {
revert AddressInsufficientBalance(address(this));
   }

   (bool success, ) = recipient.call{value: amount}("");
if (!success) {
revert FailedInnerCall();
   }
  }
function functionCall(address target, bytes memory data) internal
returns (bytes memory) {
return functionCallWithValue(target, data, 0);
  }
function functionStaticCall(address target, bytes memory data)
internal view returns (bytes memory) {
   (bool success, bytes memory returndata) = target.staticcall(data);
return verifyCallResultFromTarget(target, success, returndata);
  }
```

# INFORMATIONAL & OPTIMIZATIONS

```
function functionDelegateCall(address target, bytes memory data)
internal returns (bytes memory) {
   (bool success, bytes memory returndata) =
target.delegatecall(data);
return verifyCallResultFromTarget(target, success, returndata);
 }
library AddressCast {
 error AddressCast_InvalidSizeForAddress();
 error AddressCast_InvalidAddress();

function toBytes32(bytes calldata _addressBytes) internal pure
returns (bytes32 result) {
if (_addressBytes.length > 32) revert AddressCast_InvalidAddress();
   result = bytes32(_addressBytes);
   unchecked {
uint256 offset = 32 - _addressBytes.length;
    result = result >> (offset * 8);
  }
 }

function toBytes32(address _address) internal pure returns (bytes32
result) {
   result = bytes32(uint256(uint160(_address)));
 }
function toBytes(bytes32 _addressBytes32, uint256 _size) internal
pure returns (bytes memory result) {
if (_size == 0 || _size > 32) revert
AddressCast_InvalidSizeForAddress();
   result = new bytes(_size);
   unchecked {
```

# INFORMATIONAL & OPTIMIZATIONS

```
uint256 offset = 256 - _size * 8;
assembly {
      mstore(add(result, 32), shl(offset, _addressBytes32))
    }
   }
  }

function toAddress(bytes32 _addressBytes32) internal pure returns
(address result) {
   result = address(uint160(uint256(_addressBytes32)));
  }

function toAddress(bytes calldata _addressBytes) internal pure
returns (address result) {
if (_addressBytes.length != 20) revert
AddressCast_InvalidAddress();
   result = address(bytes20(_addressBytes));
  }
}
abstract contract Context {
  function _msgSender() internal view virtual returns (address) {
return msg.sender;
}
  function _msgData() internal view virtual returns (bytes calldata) {
return msg.data;
  }
function _contextSuffixLength() internal view virtual returns
(uint256) {
return 0;
  }
}
```

# INFORMATIONAL & OPTIMIZATIONS

```
interface IERC165 {
  function supportsInterface(bytes4 interfaceId) external view returns
(bool);
}
```

**Suggestion:**
To reduce high gas fees. It is suggested to remove unused code from the contract.

# ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

www.expelee.com

expeleeofficial

expelee

Expelee

expelee

expelee_official

expelee-co

expelee

Building the Futuristic Blockchain Ecosystem

# DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools.

Under no circumstances did Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway do your own research and protect yourselves from scams.

This document should not be presented as a reason to buy or not buy any particular token. The Expelee team disclaims any liability for the resulting losses.

expelee

Building the Futuristic Blockchain Ecosystem