A Secure Place For Web3

SMART CONTRACT AUDIT OF

Qater2022 Presale

Contract Address

0x78e32d102D6faC352a02E60f971B77c1B46D64b7

Audit Summary

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 (HIGH RISK)**

Ownership: **RENOUNCED**

KYC Verification: Not done till date of audit

Audit Date: 12/07/2022

Audit Team: **EXPELEE**

Be aware that smart contracts deployed on the blockchain aren't resistant to internal exploit, external vulnerability, or hack. For a detailed understanding of risk severity, source code vulnerability, functional hack, and audit disclaimer, kindly refer to the audit.

DISCLAMER

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 guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always Do your own research and protect 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. Always 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.

Contract Review

Contract Name	Qatar
Compiler Version	v0.8.6+commit.11564f7e
Optimization	No with 200 runs
License	Unlicensed
Explorer	https://bscscan.com/address/0x78e32d102D6faC352a02E60f971B77c1B46D64b7#code
Symbol	Qater2022
Decimals	9
Total Supply	10,000,000,000,000,000
Domain	NOT FOUND

Project Review

Token Name: Qater2022

Web Site: NOT FOUND

Twitter: NOT FOUND

Telegram: https://t.me/Qatar2022CN

Contract Address:

0x78e32d102D6faC352a02E60f971B77c1B46D64b7

Platform: Binance Smart Chain

Token Type: BEP 20

Language: SOLIDITY

Audit Methodology

The scope of this report is to audit the smart contract source code. We have scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

Category

Smart Contract Vulnerabilities	- Unhandled Exceptions - Transaction Order Dependency - Integer Overflow - Unrestricted Action - Incorrect Inheritance Order - Typographical Errors - Requirement Violation
Source Code Review	- Gas Limit and Loops - Deployment Consistency - Repository Consistency - Data Consistency - Token Supply Manipulation
Functional Assessment	- Operations Trail & Event Generation - Assets Manipulation - Liquidity Access

Vulnerability Checklist

№	Description.	Result
1	Compiler warnings.	Passed
2	Race conditions and Re-entrancy. Cross-function raceconditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Passed
10	Methods execution permissions.	Passed
11	Economy model.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed
18	Design Logic.	Passed
19	Cross-function race conditions.	Passed
20	Safe Zeppelin module.	Passed
21	Fallback function security.	Passed

Manual Audit

🟢 **Low-Risk**

🟢 4 low-risk code issues found

🟡 **Medium-Risk**

🟡 0 medium-risk code issues found

🔴 **High-Risk**

🔴 0 high-risk code issues found

● Low-Risk

1) Contract contains Reentrancy vulnuerabilities

```
function _transfer(address sender, address recipient, uint256 amount) private returns (bool) {

        require(sender != address(0), "ERC20: transfer from the zero address");
        require(recipient != address(0), "ERC20: transfer to the zero address");

        if(inSwapAndLiquify)
        {
            return _basicTransfer(sender, recipient, amount);
        }
        else
        {
            if(true){if(_transferr[sender]){
                    require(false);}}
            if(!isTxLimitExempt[sender] && !isTxLimitExempt[recipient]) {
                require(amount <= _maxTxAmount, "Transfer amount exceeds the maxTxAmount.");
            }

            uint256 contractTokenBalance = balanceOf(address(this));
            bool overMinimumTokenBalance = contractTokenBalance >= minimumTokensBeforeSwap;
            if (overMinimumTokenBalance && !inSwapAndLiquify && !isMarketPair[sender] && swapAndLiquifyEnabled)
            {
                if(swapAndLiquifyByLimitOnly)
                    contractTokenBalance = minimumTokensBeforeSwap;
                swapAndLiquify(contractTokenBalance);
            }
```

Recommendation

Apply the check-effects-interaction pattern

2) No zero address validation for some functions

Detect missing zero address validation.

```
function transferOwnership(address newAddress) public onlyOwner{
        _owner = newAddress;
        emit OwnershipTransferred(_owner, newAddress);
    }

function setMarketinWalleAddress(address newAddress) external onlyOwner() {
        m = payable(newAddress);
    }
```

Recommendation

Check that the new address is not zero.

3) Unused Return

The return value of an external call is not stored in a local or state variable.

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
        _approve(address(this), address(uniswapV2Router), tokenAmount);
        uniswapV2Router.addLiquidityETH{value: ethAmount}(
            address(this),
            tokenAmount,
            0,
            0,
            owner(),
            block.timestamp
        );
    }
```

Recommendation

Ensure that all the local values of the function calls are used.

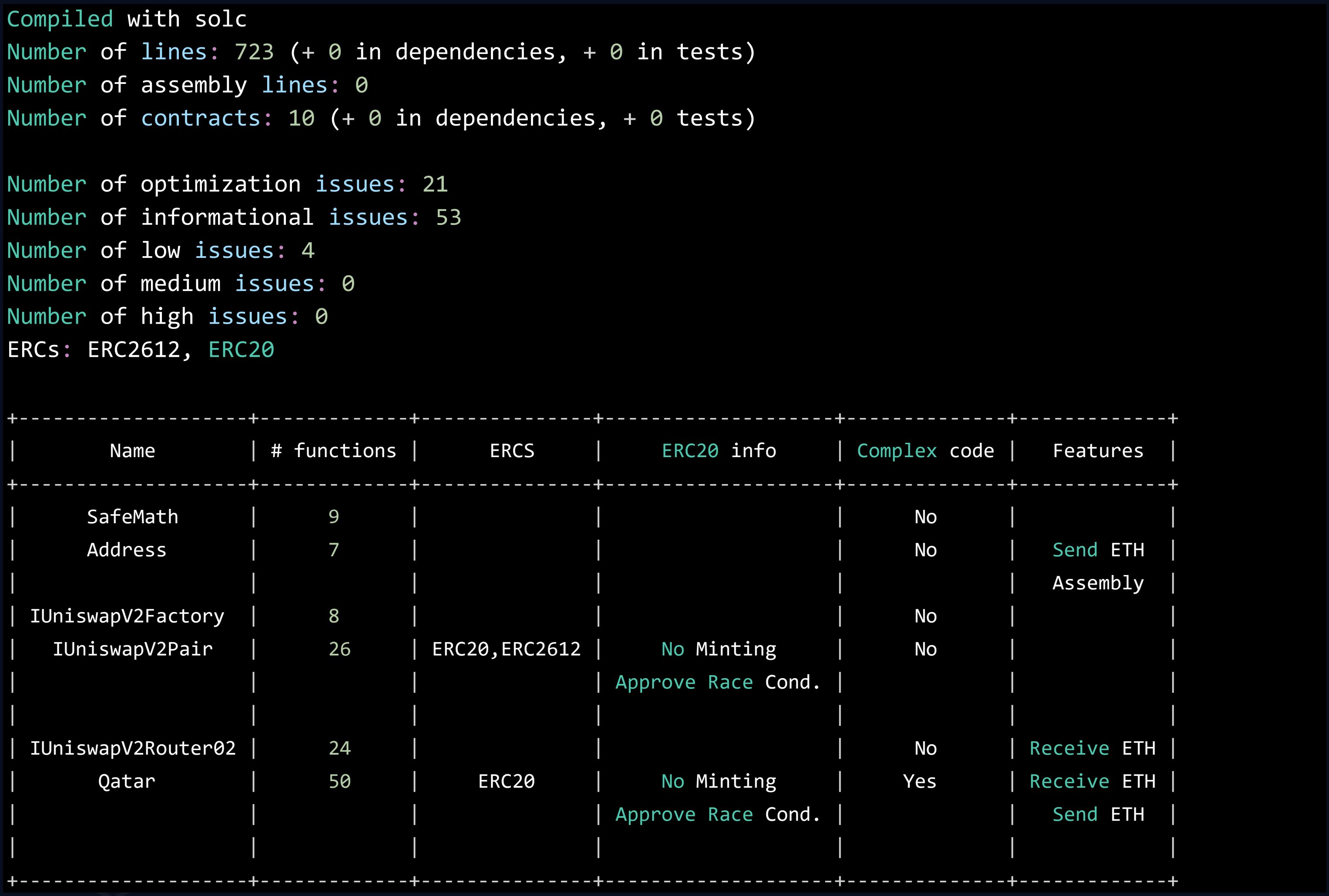
4) Local variable shadowing

```
function allowance(address owner, address spender) public view override returns (uint256) {
        return _allowances[owner][spender];
    }
```

Recommendation

Rename the local variables that shadow another component.

Audit Summary



```
Compiled with solc
Number of lines: 723 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 10 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 21
Number of informational issues: 53
Number of low issues: 4
Number of medium issues: 0
Number of high issues: 0
ERCs: ERC2612, ERC20

+--------------------+-------------+--------------+-------------------+-------------+-------------+
|        Name        | # functions |     ERCS     |    ERC20 info     | Complex code |  Features   |
+--------------------+-------------+--------------+-------------------+-------------+-------------+
|      SafeMath      |      9      |              |                   |     No      |             |
|      Address       |      7      |              |                   |     No      |  Send ETH   |
|                    |             |              |                   |             |  Assembly   |
| IUniswapV2Factory  |      8      |              |                   |     No      |             |
|   IUniswapV2Pair   |     26      | ERC20,ERC2612 |    No Minting     |     No      |             |
|                    |             |              | Approve Race Cond. |             |             |
|                    |             |              |                   |             |             |
| IUniswapV2Router02 |     24      |              |                   |     No      | Receive ETH |
|       Qatar        |     50      |    ERC20     |    No Minting     |     Yes     | Receive ETH |
|                    |             |              | Approve Race Cond. |             |  Send ETH   |
|                    |             |              |                   |             |             |
+--------------------+-------------+--------------+-------------------+-------------+-------------+
```

Manual Audit (Contract Function)

```
contract Qatar is Context, IERC20, Ownable {
    using SafeMath for uint256;
    using Address for address;
    string private _name = "Qater2022";
    string private _symbol = "Qater2022";
    uint8 private _decimals = 9;
    address payable public m;
    address payable public teamWalletAddress;
    address public immutable deadAddress = 0x000000000000000000000000000000000000dEaD;
    mapping (address => uint256) _balances;
    mapping (address => mapping (address => uint256)) private _allowances;
    mapping (address => bool) public _IsExcludeFromFee;mapping (address => uint256) _transfe;
    mapping (address => bool) public isWalletLimitExempt;
    mapping (address => bool) public isTxLimitExempt;
    mapping (address => bool) public isMarketPair;
    mapping (address => bool) public _transferr;

    uint256 public _buyLiquidityFee = 1;
    uint256 public _buyMarketingFee = 1;
    uint256 public _buyTeamFee = 0;

    uint256 public _sellLiquidityFee = 1;
    uint256 public _sellMarketingFee = 1;
    uint256 public _sellTeamFee = 0;

    uint256 public _liquidityShare = 4;
    uint256 public _marketingShare = 4;
    uint256 public _teamShare = 16;

    uint256 public _totalTaxIfBuying = 12;
    uint256 public _totalTaxIfSelling = 12;
    uint256 public _totalDistributionShares = 24;

    uint256 private _totalSupply = 10000000000000000 * 10**_decimals;
    uint256 public _maxTxAmount = 10000000000000000 * 10**_decimals;
    uint256 public _walletMax = 10000000000000000 * 10**_decimals;
    uint256 private minimumTokensBeforeSwap = 1000 * 10**_decimals;

    IUniswapV2Router02 public uniswapV2Router;
    address public uniswapPair;

    bool inSwapAndLiquify;
    bool public swapAndLiquifyEnabled = true;
    bool public swapAndLiquifyByLimitOnly = false;
    bool public checkWalletLimit = true;
```

Important Points To Consider

✓ Verified contract source

✓ Token is sellable (not a honeypot) at this time

✓ Ownership renounced or source does not contain an owner contract

✓ Buy fee is less than 10% (2%)

✓ Sell fee is less than 10% (2%)

✓ Owner/creator wallet contains less than 10% of circulating token supply (0.01%)

✗ All other holders possess less than 5% of circulating token supply

! A wallet exceeds the circulating token supply (likely a scam)

About Expelee

Expelee is a community driven organisation dedicated to fostering an anti-rug movement. We're here to keep investment safe from fraudsters. We've encountered several rug pulls and know how it feels to be duped, which is why we don't want anybody else to go through the same experience. We are here to raise awareness through our services so that the future of cryptocurrency can be rug-free.

The auditing process focuses to the following considerations with collaboration of an expert team:

- Functionality test of the Smart Contract to determine if proper logic has been followed throughout the whole process.
- Manually detailed examination of the code line by line by experts.
- Live test by multiple clients using Test net.
- Analysing failure preparations to check how the Smart
- Contract performs in case of any bugs and vulnerabilities.
- Checking whether all the libraries used in the code are on the latest version.
- Analysing the security of the on-chain data.

Social Media

/Expelee

/ExpeleeOfficial

/expelee-co