expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

ALBETROS

TOKEN OVERVIEW

Risk Findings

Sever ty	Found
High	0
Medium	0
Low	0
Informational	0

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 Burn Tokens ?	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	**No Risk**
Audit Date	**29 March 2024**

CONTRACT DETAILS

Token Address: 0xB087CB524daEF1CdAC5E78b7b9F2Ed20B891C187

Name: ALBETROS

Symbol: ARS

Decimals: 18

Network: BscScan

Token Type: BEP-20

Owner: 0xC978f05787a944DDb0602AF825057B7Ee081a38e

Deployer: 0xC978f05787a944DDb0602AF825057B7Ee081a38e

Token Supply: 100 Billion

Checksum: Ae1c3a4fbb6e83e8393a57617b5a5132

Testnet:
https://testnet.bscscan.com/address/0x0970d6fbaee1ffadc1d7ec7ae92ec187b6264a2a#code

expelee

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

Des gn Log c	Passed
Comp ler warn ngs	Passed
Pr vate user data leaks	Passed
T mestamps dependence	Passed
Integer overflow and underflow	Passed
Race cond t ons & reentrancy. Cross-funct on race cond t ons	Passed
Poss ble delays n data del very	Passed
Oracle calls	Passed
Front Runn ng	Passed
DoS w th Revert	Passed
DoS w th block gas l m t	Passed
Methods execut on perm ss ons	Passed
Economy model	Passed
Impact of the exchange rate on the log c	Passed
Mal c ous event log	Passed
Scop ng and declarat ons	Passed
Un n t al zed storage po nters	Passed
Ar thmet c accuracy	Passed
Cross-funct on race cond t ons	Passed
Safe Zeppl n module	Passed

RISK CLASSIFICATION

When perform ng smart contract aud ts, our spec al sts look for known vulnerab l t es as well as log cal and acces control ssues w th n the code. The explo tat on of these ssues by mal c ous actors may cause ser ous f nanc al damage to projects that fa led to get an aud t n t me. We categor ze these vulnerab l t es by the follow ng 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

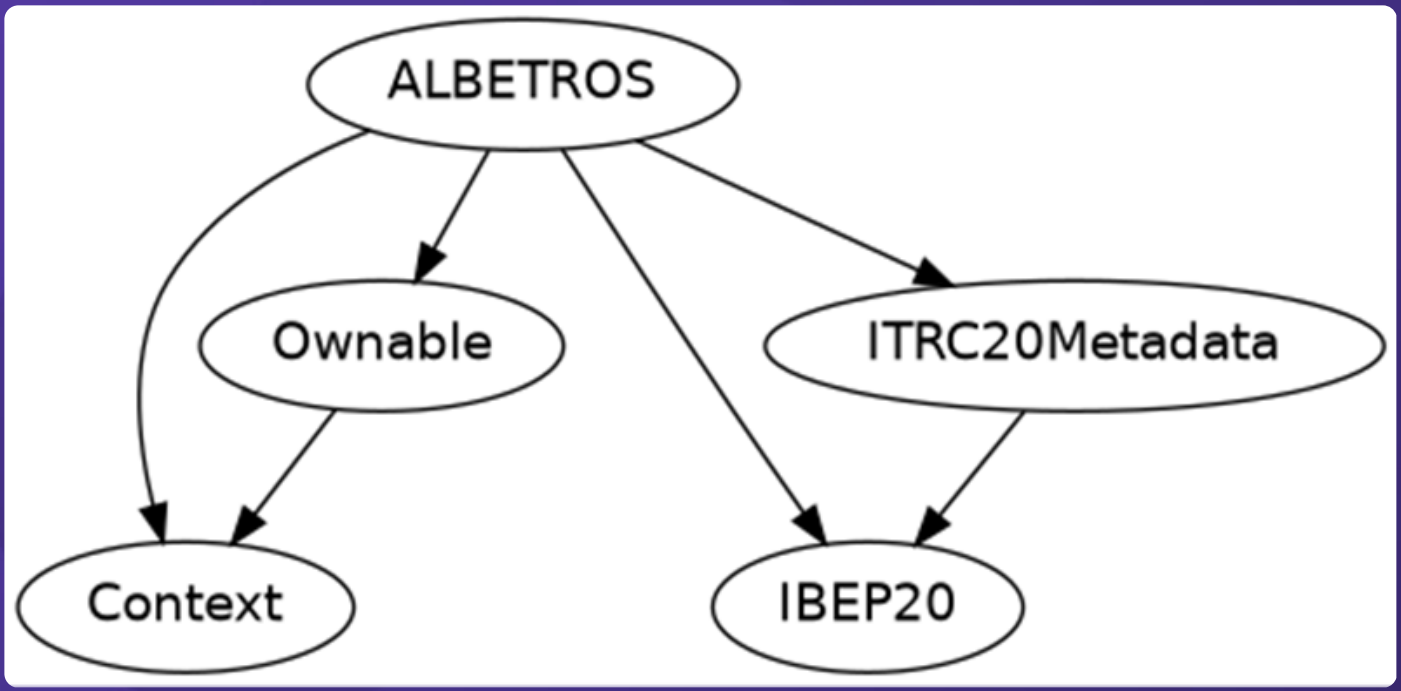
Issues on this level are minor details and warning that can remain unfixed.

Informational

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

expelee

INHERITANCE TREE



STATIC ANALYSIS

```
INFO:Detectors:
ALBETROS.allowance(address,address).owner (ALBETROS.sol#133) shadows:
        - Ownable.owner() (ALBETROS.sol#51-53) (function)
ALBETROS._approve(address,address,uint256).owner (ALBETROS.sol#281) shadows:
        - Ownable.owner() (ALBETROS.sol#51-53) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Context._msgData() (ALBETROS.sol#13-15) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (ALBETROS.sol#6) allows old versions
solc-0.8.24 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
ALBETROS.constructor() (ALBETROS.sol#92-97) uses literals with too many digits:
        - _mint(msg.sender,100000000000 * 10 ** 18) (ALBETROS.sol#96)
ALBETROS.slitherConstructorConstantVariables() (ALBETROS.sol#76-313) uses literals with too many digits:
        - MAX_SUPPLY = 100000000000 * 10 ** 18 (ALBETROS.sol#87)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
ALBETROS._decimals (ALBETROS.sol#85) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:ALBETROS.sol analyzed (5 contracts with 93 detectors), 8 result(s) found
```

TESTNET VERSION

1- Approve (passed):
https://testnet.bscscan.com/tx/0xdc9da3afbd572a6fc9498a977affe29b894bde54a3a54d2001984f1ca4d7f7bb

2- Increase Allowance (passed):
https://testnet.bscscan.com/tx/0x7696de5c6921593498a465df225dd9653f7ab91fac05d6663c82a57d6e1cf99e

3- Decrease Allowance (passed):
https://testnet.bscscan.com/tx/0x19f8a2cef33f2c41c8bfab38d2d166797fab1ef52e1c7731db5e6b165124ed5f

4- M nt (passed):
https://testnet.bscscan.com/tx/0xfb2efac9ec9509506414bb64b92d9a303c336bde9279d331afa706094b2bde9b

5- Burn (passed):
https://testnet.bscscan.com/tx/0x86c2589a2974c760d0b6239bbcdce828585e1185a6a6978bf9ff86b58af69a56

6- Transfer Ownersh p (passed):
https://testnet.bscscan.com/tx/0xdb36aaca02e324474cb0b1fdc9433038fc61ae164d66ab2c634881940320451f

MANUAL REVIEW

Sever ty Cr ter a

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			

HİGH RISK FINDING

Centralization – Owner can Burn Tokens. Severity: Low
Function: burn
Status: Open

Overview:
The owner can burn tokens without approval from any wallet.

```
function burn(address account, uint256 value) public onlyOwner {
 _burn(account, value);
 }
```

Suggestion:
There should not be any burning without any allowance from the user.

HİGH RISK FINDING

Centralization – Owner Can Mint Tokens. Severity: Low

Status: Open

Function: mint

Overview:

The owner is able to mint unlimited tokens which is not recommended as this functionality can cause the token to lose it's value and the owner can also use it to manipulate the price of the token.

```
function mint(address account, uint256 value) public onlyOwner {
 _mint(account, value);
 }
```

Suggestion:

It is recommended that the total supply of the token should not be changed after initial deployment.

LOW RISK FINDING

Centralization – Local Variable
Shadowing Severity: Low
Status: Open
Function: _approve and allowance

Overview:

```
function allowance(address owner, address spender) public view virtual override returns (uint256) {
 return _allowances[owner][spender];
  }
function _approve(
 address owner,
 address spender,
 uint256 amount
 ) internal virtual {
 require(owner != address(0), "BEP20: approve from the zero address");
 require(spender != address(0), "BEP20: approve to the zero address");

 _allowances[owner][spender] = amount;
 emit Approval(owner, spender, amount);
  }
```

Suggestion:
Rename the local variable that shadows another component.

INFORMATIONAL & OPTIMIZATIONS

Optimization

Severity: Informational

Subject: Floating Pragma.

Status: Open

Overview:

It is considered best practice to pick one compiler version and stick with it. With a floating pragma, contracts may accidentally be deployed using an outdated.

```
pragma solidity ^0.8.0;
```

Suggestion:

Adding the latest constant version of solidity is recommended, as this prevents the unintentional deployment of a contract with an outdated compiler that contains unresolved bugs.

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 _msgData() internal pure returns (bytes calldata) {
 return msg.data;
  }
```

ABOUT EXPELEE

Expelee s a product-based asp rat onal Web3 start-up. Cop ng up w th numerous solut ons for blockcha n secur ty and construct ng a Web3 ecosystem from deal mak ng platform to developer host ng open platform, wh le also develop ng our own commerc al and susta nable blockcha n.

www.expelee.com

expeleeofficial

Expelee

expelee_official

expelee

expelee

expelee-co

Building the Futuristic Blockchain Ecosystem

DISCLAIMER

All the content prov ded n th s document s for general nformat on only and should not be used as f nanc al adv ce or a reason to buy any nvestment. Team prov des no guarantess aga nst the sale of team tokens or the removal of l qu d ty by the project aud ted n th s document.

Always do your own research and project yourselves from be ng scammed. The Expelee team has aud ted th s project for general nformat on and only expresses the r op n on based on s m lar projects and checks from popular d agnost c tools.

Under no c rcumstances d d Expelee rece ve a payment to man pulate those results or change the award ng badge that we w ll be add ng n our webs te. Alway do your own research and protect yourselves from scams.

Th s document should not be presented as a reason to buy or not buy any part cular token. The Expelee team d scla ms any l ab l ty for the result ng losses.

Building the Futuristic Blockchain Ecosystem