2 Dec 2022

DRIVENsecurity
Premium Audit

Ark Fi

Static Code Analysis & Manual
Verification For Smart Contract

WWW.DRIVENECOSYSTEM.COM

Disclaimer

Accepting a project audit can be viewed as a sign of confidence and is typically the first indicator of trust for a project, but it does not guarantee that a team will not remove all liquidity, sell tokens, or engage in any other type of fraud. There is also no method to restrict private sale holders from selling their tokens. It is ultimately your obligation to read through all documentation, social media posts, and contract code for each particular project in order to draw your own conclusions and define your own risk tolerance.

DRIVENlabs Inc. accepts no responsibility for any losses or encourages speculative investments. This audit's material is given solely for information reasons and should not be construed as investment advice.

Table Of Contents

Project Details 3

Static Analysis 4

Issues 6

Recommendations 8

Conclusion 9

Project Details

Name of the project:
Ark Fi

Website:
https://www.arkfi.io/

Type of the Smart Contract:
Custom ERC721 Token

Chain:
Binance Smart Chain

Address:
0x2222223B05B5842c918a868928F57cD3A0332222

Explorer Link:
https://bscscan.com/address/0x2222223B05B5842c918a868928F57cD3A0332222#code

Static Analysis

SWC ISSUES	STATUS
Function Default Visibility	PASSED
Integer Overflow and Underflow	PASSED
Outdated Compiler Version	PASSED
Floating Pragma	PASSED
Unchecked Call Return Value	PASSED
Unprotected Ether Withdrawal	PASSED
Unprotected SELFDESTRUCT Instruction	PASSED
Reentrancy	HIGH SEVERITY (SOLVED)
State Variable Default Visibility	LOW SEVERITY (SOLVED)
Uninitialized Storage Pointer	PASSED
Assert Violation	PASSED
Use of Deprecated Solidity Functions	PASSED
Delegatecall to Untrusted Callee	PASSED
DoS with Failed Call	PASSED
Transaction Order Dependence	PASSED
Authorization through tx.origin	PASSED
Block values as a proxy for time	PASSED
Signature Malleability	PASSED
Incorrect Constructor Name	PASSED
Shadowing State Variables	PASSED
Weak Sources of Randomness from Chain Attributes	PASSED
Missing Protection against Signature Replay Attacks	PASSED
Lack of Proper Signature Verification	PASSED
Requirement Violation	PASSED

Static Analysis

SWC ISSUES	STATUS
Write to Arbitrary Storage Location	PASSED
Incorrect Inheritance Order	PASSED
Insufficient Gas Griefing	PASSED
Arbitrary Jump with Function Type Variable	PASSED
DoS With Block Gas Limit	PASSED
Typographical Error	PASSED
Right-To-Left-Override control character (U+202E)	PASSED
Presence of unused variables	PASSED
Unexpected Ether balance	PASSED
Hash Collisions With Multiple Variable Length Arguments	PASSED
Message call with hardcoded gas amount	PASSED
Code With No Effects	PASSED

Issues

Static Code Analysis

- SWC 108: It is best practice to set the visibility of state variables explicitly:
  uint256 maxCwrWithoutNft = 1500;

- SWC 108: It is best practice to set the visibility of state variables explicitly:
  uint256 openingHour = type(uint256).max;

Manual Verification

Compilation Errors:

- ARK is not declared before is used in the “setVaultAddress” function:

```
function setVaultAddress(address vaultAddress) external onlyCEO {
            ...
              IBEP20(>>> ARK).approve(address(vault), type(uint256).max);
              >>>> ARK not declared
            ...
  }
```

- ”getClaimableRewards” function should return an uint256 variable

```
          function getClaimableRewards(address investor) public view
returns(uint256) {
    ...
        if(claimedAlready >= totalRewardsPerShare * shares[id]) return;  <<<
        >>>> Return argument (uint256) required
    ...
    }
```

Issues

Violation of the Checks-Effects-Interactions pattern. The “_claim” function is used in the following functions: “levelUp”, “_transfer”, “claimRewards” and “claimRewardsFor”.

```
function _claim(address investor) internal {
      if(balanceOf(investor) == 0) return;
      uint256 id = tokenOfOwnerByIndex(investor, 0);
      uint256 claimedAlready = excluded[id];
      if(claimedAlready >= totalRewardsPerShare * shares[id]) return;
      uint256 claimableNow = shares[id] * (totalRewardsPerShare-
      claimedAlready) / veryBigNumber;

      >>> BUSD.transfer(investor, claimableNow);

      claimedRewards[id] += claimableNow;
      excluded[id] = totalRewardsPerShare;

      >>> Move it there

      emit RewardsClaimed(investor, claimableNow);
  }
```

- The "mintToWalletPaid(address to, uint256 level)" function is a function that allows the "CEO" to mint an NFT to the "to" address. This function cannot be called if the CEO has more than 1 NFT in his wallet, which is not a proper implementation according to how the smart contract should work:

```
require(balanceOf(msg.sender) == 0, "Max NFT per wallet exceeded");
```

Instead of this require statement, use:

```
require(balanceOf(to) == 0, "Max NFT per wallet exceeded");
```

Recommendations

- For the "lockNft()" function add the following require statement

```
require(locked[id] != true, "NFT already locked!");
```

- For the "unlockNft()" function add the following require statement

```
require(locked[id] != false, "NFT already unlocked!");
```

- Add a “require” statement for BUSD transfers in the following functions: “_claim”, “mint”, “mintToWalletPaid”, “levelUp”, “buy”, “addToRewards”.

```
BUSD.transfer(investor, claimableNow);
```

TO

```
require(BUSD.transfer(investor, claimableNow), “Failed!”);
```

Conclusion

Done: The Ark Fi team is aware of our findings.

Done [06 Dec. 2022]: Solve the issues.

Update [17 Dec. 2022]: Issues were solved and the smart contract was deployed.

2 Dec 2022

Thank you!