Cyberscope

# Audit Report

# Altair Factory

November 2022

SHA256 09dfd6013ceafce40f667e5ae980d72a510fa9df6f44425bde3f18d8db5ade8c

Audited by © cyberscope

# Table of Contents

# Contract Review

| | |
|---|---|
| **Contract Name** | AltairFactory |
| **Compiler Version** | v0.8.14+commit.80d49f37 |
| **Testing Deploy** | https://testnet.bscscan.com/token/0x068240cf4f9Ed80452d99001eeaB82fd19c22Cbf |
| **Domain** | https://nali.finance |

# Audit Updates

| | |
|---|---|
| **Initial Audit** | 16th November 2022 |
| **Corrected** | |

# Source Files

| Filename | SHA256 |
| --- | --- |
| **@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol** | da66c17044345dc892d85bd7ddc9745d25df0b3dacfba8f84eb87c60d6e40fe3 |
| **@openzeppelin/contracts-upgradeable/proxy/ClonesUpgradeable.sol** | 11d95e1dddcd4351da5eb6eb4b1d6fa0f30afc45e4bad6d34028185a28a1cf65 |
| **@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol** | cd823c76cbf5f5b6ef1bda565d58be66c843c37707cd93eb8fb5425deebd6756 |
| **@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol** | c05b019a0b3bee8f3fac2da7c929f7d665b97d6d046aa35126615fff11205119 |
| **@openzeppelin/contracts-upgradeable/token/ERC1155/ERC1155Upgradeable.sol** | 7b5314b4e3ddc497ba6bb51c783cbc38762526cd7e801fbc28ba9e00317c2a76 |
| **@openzeppelin/contracts-upgradeable/token/ERC1155/extensions/ERC1155BurnableUpgradeable.sol** | 80c8ff8a46a2b197bee6518186627db7dda386b4a840aa162d32257bc97c9fff |
| **@openzeppelin/contracts-upgradeable/token/ERC1155/extensions/ERC1155SupplyUpgradeable.sol** | 98d3570efe134810096cab2ce5d39c51323d9216e8528198fac44dd15cd3c841 |

| | |
|---|---|
| **@openzeppelin/contracts-upgradeable/token/ERC1155/extensions/ERC1155URIStorageUpgradeable.sol** | 7da34d679483559e38cbae26e66e21d39b4e5f191b1942669b45ac29cd965f09 |
| **@openzeppelin/contracts-upgradeable/token/ERC1155/extensions/IERC1155MetadataURIUpgradeable.sol** | 61e3317af2516530f091665d198fc3e27fb514038600fb07b7813913f439775f |
| **@openzeppelin/contracts-upgradeable/token/ERC1155/IERC1155ReceiverUpgradeable.sol** | 7108589dbf9528c2ffe4f17fe489e8183be55c15b51af3b88c70252c13bac539 |
| **@openzeppelin/contracts-upgradeable/token/ERC1155/IERC1155Upgradeable.sol** | 5321f224c08b59651968dde0db5abeec4ea0bdf371c7b0c25707e56b455882fe |
| **@openzeppelin/contracts-upgradeable/token/ERC1155/utils/ERC1155HolderUpgradeable.sol** | fe3758383dfee87be40050cc903408f0a4c7f20eb430931dcbb8fa337cbdfa4f |
| **@openzeppelin/contracts-upgradeable/token/ERC1155/utils/ERC1155ReceiverUpgradeable.sol** | 59bd51ef07cf35a5de2e0090231ed4398d529173a2749c04fb04874613b3da50 |
| **@openzeppelin/contracts-upgradeable/token/ERC20/extensions/draft-IERC20PermitUpgradeable.sol** | b97515a88e75c313eacf0a27c9439ef371d86d4c2730d3b13076640942f813df |

| | |
|---|---|
| **@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol** | 4e09a7479aa3e7c313f8fc141c4c8fc04e0abfeb8754615ef7d78ec94c298b07 |
| **@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol** | b7410d275fc7d26e36b0851541d6ff290593ba72d64b5c906978124b123915c1 |
| **@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol** | 35fb271561f3dc72e91b3a42c6e40c2bb2e788cd8ca58014ac43f6198b8d32ca |
| **@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol** | 5fb301961e45cb482fe4e05646d2f529aa449fe0e90c6671475d6a32356fa2d4 |
| **@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol** | fd84e5284eccc479268f0ef36b830019d4f7999ceb7959430d8d8d9e602dd4ef |
| **@openzeppelin/contracts-upgradeable/utils/introspection/IERC165Upgradeable.sol** | a39bc026ad6214e9ecd526bd4a1ddf9862d80bd4a9d0d031d9bafa4c3c147c0b |
| **@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol** | e7b950eee23563e23989a3b51a1456614a1838084eef1fad04eb2be0bc280f48 |
| **@openzeppelin/contracts-upgradeable/utils/structs/EnumerableSetUpgradeable.sol** | 014846dc6c387e8fd6d31df636c28898eed601dc668725edbb1a0606c58d4b7e |

| | |
|---|---|
| **contracts/AltairFactory.sol** | 09dfd6013ceafce40f667e5ae980d72a510fa9df6f44425bde3f18d8db5ade8c |
| **contracts/interfaces/IAltairFund.sol** | 30f415f676820b6f9f005e1022eb7c23ac5c7e211640919f2c2bf9a71a64dd98 |
| **contracts/interfaces/IAltairMaster.sol** | d441fc5c3c90f5557ca8715269d3fc23f9aaae007ff8954dd61a5f9d8599cd25 |
| **contracts/interfaces/IAltairShop.sol** | e4bbaca996c77e95097b2641444dbb32a777067b60e23b763c3fe7f7b48b8c0b |
| **contracts/interfaces/IAltairSwap.sol** | 0b2a9ce80407df392c551deb6179f0031a9d69a0a227622c26bd68eaa925b7f1 |
| **contracts/interfaces/IAltairSwapInfo.sol** | dc46d60af278bfd40dce138d166d2dde275678009060baa3c9851bd68cfe60d7 |
| **contracts/interfaces/IContractInitializer.sol** | 7606aa126257e99dca9aec4b61997d5886dd0b4960a04c0b710cf6dc5c5d373a |
| **contracts/interfaces/INaliToken.sol** | b516894970890c70ee7c748285b79adca4c45c34e6f6822a1de08b1843ba14f6 |
| **contracts/interfaces/IProxyCall.sol** | bd6f89340af3460c079dd3853d2e1a87c1e02ae124b3b732b83ef3b32deac6bd |
| **contracts/interfaces/IRoles.sol** | d13aac5175ddfd3af1ee4dc652d46b5f317214c0455a10ee580c0b2d414876e0 |
| **contracts/interfaces/ITreasury.sol** | 23176edee2b0b416db08b8bd686821cd51afdeb015f8e97a85a83141b8134315 |
| **contracts/interfaces/IWETH.sol** | 4b36d18ea8a606a912484eef60d6e2c5e4c3baa8ac64c3541775394fa1fd633e |
| **contracts/libs/AltairLib.sol** | 95ae3593a0abb118bb7cf4670fb28b119873b8a7861ceb6cd39d9b099bb3bb30 |
| **contracts/libs/SwapData.sol** | c4b79ab1f23186ae4c19b628a5cb4019c5308c109bb707f553360ab3ff06ca46 |
| **contracts/NaliAltair.sol** | 49e87eec12e78e062fa10e576836fce951 |

| | |
|---|---|
| | 82577c4ca430817e2643721f3eb6dd |
| **contracts/AltairFund.sol** | a3ea8e8bc03550099158c91b5014522191d9188897e06963eeff937cf99a2d1b |
| **contracts/libs/TransferHelper.sol** | 5c72ac8713b20401a93862747e1ea2e574845b808f4446078bb13731e9d611dc |
| **contracts/libs/HomoraMath.sol** | ca67e5cdf097ed2f62fc3d206b3ecd98e4255a9f3de350ff492f98cd5e926406 |
| **contracts/AltairMaster.sol** | e7f8e0ca801fd8f7fedf2bda28211f973cf350ad646881974770e7bc8b6291be |

# Introduction

The Altair Factory ecosystems contracts are implemented as an upgradable proxy. It consists of AltairFund, AltairMaster, and AltairFactory contracts.

## AltairFactory

The Altair Factory is responsible for creating funds, and managing the platform's parameters. Additionally, the contract is responsible for minting, burning Nali NFTs, and subscribing to an AltainFund.

## AltairFund

The Altair Factory implements a pooling mechanism for the created funds. Users can subscribe to a fund, redeem, and emergencyRedeem their investment.

## AltairMaster

The AltairMaster is responsible for providing information from the world outside to the blockchain. To be more specific the AltairMaster contract keeps info about tokens prices from the liquidity pool. Additionally, the contract provides price data from three different Oracles. Chainlinks, Band, and 1-inch oracle.

# Roles

The contract roles are provided by an outside source. The Roles contract is out of the scope of this audit.

## AltairFactory

The AltairFactory contract has an admin role.

### Admin

The admin has the authority to

- Change CustomToken address.
- Change Native address.
- Change NaliToken address.
- Change subscription fee.
- Change Redeem fee.
- Change fee variable.
- Set UseSwapInfo.
- Change Manager fee.
- Set Treasury address.
- Set Master address.
- Set Shop address.
- Set AltairSwap address.
- Set SwapContract address.
- Change Monthly cost.
- Change the Free Trial period.
- Update contract implementation.
- Update proxy call contract
- Update Roles contract.

## AltairMaster

The AltairMaster contract has an admin role.

### Admin

The admin has the authority to

- Change the router address.
- Update tokens name.
- Add PancakePriceToken.
- Update Supporting Fee On TransferTokens.

# AltairFund

The AltairFund contract has a manager role and a factory role.

## Manager

The manager role consists of the managerOwner and the authorized users. The manager role has the authority to

- Set copyTrading.
- Set DCA.
- Set Manager monthly fee.
- Redeem the manager's monthly fee.
- Pause or Unpause funds subscriptions.
- Create TargetNames.
- Update rebalance period.
- Update manager property.
- Update Non-Balance Manager Names
- Rebalance.
- Can swap non-index tokens to BNB.
- Set Authorized address.

## Factory

The factory roles have the authority to

- Update the platform address.
- Update max manager monthly fee.

# Contract Healthy Functionality ⚠️

The contract is not well-structured and contains a lot of inaccuracies. We kindly advise the team to well-form their requirements and re-write the contract.

Following the above-mentioned note, we will mention some samples of the code segments that depict the inaccuracies.

### Redundant Statements and Permission issues

```
function updateAllPlatformAddresses(address[] memory fundsArray)
    external
    returns (bool)
{
    fundsArray = funds.values();

    uint256 fundsLength = fundsArray.length;

    if (fundsArray.length == 0) {
        fundsArray = new address[](fundsLength);

        for (uint256 i; i < fundsLength; i++) {
            fundsArray[i] = funds.at(i);
            IAltairFund(fundsArray[i]).updatePlatformAddresses();
        }
    } else {
        for (uint256 i; i < fundsArray.length; i++) {
            IAltairFund(fundsArray[i]).updatePlatformAddresses();
        }
    }

    return true;
}
```

### Unused State Variables

```
masterOwner = IAltairFactory(_factoryOwner).fundMaster();
altairSwap = IAltairSwap(IAltairFactory(_factoryOwner).altairSwap());
managerFeeBps = IAltairFactory(_factoryOwner).managerFee();
fundMaster = IAltairMaster(masterOwner);
factory = IAltairFactory(_factoryOwner);
platformFee = IAltairFactory(_factoryOwner).fee();
platformWallet = IAltairFactory(_factoryOwner).treasury();
```

### Configuration Issues

For instance, if the variables are not properly configured then some of the expressions may revert. In the example bellow if the _totalSupply is grater than totalUnitAfter, the expression will revert.

```
function _getNewFundUnits(
```

```
        uint256 _totalFundB4,
        uint256 _totalValueAfter,
        uint256 _totalSupply
    ) internal pure returns (uint256) {
        if (_totalValueAfter == 0) return 0;
        if (_totalFundB4 == 0) return _totalValueAfter;
        uint256 totalUnitAfter = _totalValueAfter.mul(_totalSupply).div(
            _totalFundB4
        );
        uint256 mintUnit = totalUnitAfter.sub(_totalSupply);
        return mintUnit;
    }
```

## Redundant Statements

```
require(
    !_isSmallSubs(fundvalue, totalSubs),
    "under_1%_of_fund_value"
);
if (!_isSmallSubs(fundvalue, totalSubs)) {...
```

# Contract Diagnostics

Critical Medium Minor / Informative

| Severity | Code | Description | Status |
|---|---|---|---|
| ● | VAP | Volatile Address Prediction | Unresolved |
| ● | BLC | Business Logic Concern | Unresolved |
| ● | ELFM | Exceeds Fees Limit | Unresolved |
| ● | STC | Succeeded Transfer Check | Unresolved |
| ● | MC | Missing Check | Unresolved |
| ● | MTP | Misuse of Transaction Property | Unresolved |
| ● | L02 | State Variables could be Declared Constant | Unresolved |
| ● | L04 | Conformance to Solidity Naming Conventions | Unresolved |
| ● | L09 | Dead Code Elimination | Unresolved |
| ● | L11 | Unnecessary Boolean equality | Unresolved |
| ● | L13 | Divide before Multiply Operation | Unresolved |
| ● | L14 | Uninitialized Variables in Local Scope | Unresolved |

# VAP - Volatile Address Prediction

| | |
|---|---|
| **Criticality** | Minor / informative |
| **Location** | contract.sol/AltairFactory.sol#L583 |
| **Status** | Unresolved |

## Description

The address prediction is volatile on time. Hence the function predictAddress will yield the correct prediction only in the timeframe of one second.

```
function predictAddress(address creator)
        external
        view
        returns (address naliAddress)
    {
        uint256 nonce = uint256(
            keccak256(abi.encodePacked(msg.sender, block.timestamp))
        );
        naliAddress = implementation.predictDeterministicAddress(
            _getSalt(creator, nonce + 1)
        );
    }
```

## Recommendation

The contract could predict address with alternative noonce.

# BLC - Business Logic Concern

| | |
|---|---|
| **Criticality** | critical |
| **Location** | contract.sol/AltairFactory.sol#L451 |
| **Status** | Unresolved |

## Description

The business logic seems peculiar. The implementation may not follow the expected behavior. The function updateAllMaxManagerMonthlyFee is public. Hence, the fee could be manipulated by anyone.

```
function updateAllMaxManagerMonthlyFee(
    address[] memory fundsArray,
    uint256 _fee
) external returns (bool) {
    fundsArray = funds.values();
    uint256 fundsLength = fundsArray.length;
    if (fundsArray.length == 0) {
        fundsArray = new address[](fundsLength);
        for (uint256 i; i < fundsLength; i++) {
            fundsArray[i] = funds.at(i);
            IAltairFund(fundsArray[i]).updateManagerMaxMonthlyFee(_fee);
        }
    } else {
        for (uint256 i; i < fundsArray.length; i++) {
            IAltairFund(fundsArray[i]).updateManagerMaxMonthlyFee(_fee);
        }
    }
    return true;
}
```

## Recommendation

The team is advised to carefully check if the implementation follows the expected business logic.

# ELFM - Exceeds Fees Limit

| | |
|---|---|
| **Criticality** | medium |
| **Location** | contract.sol/AltairFactory.sol#L235,255,269,906<br>contract.sol/AltairFund.sol#L211 |
| **Status** | Unresolved |

## Description

The factory contract has the authority to increase fees over 10000. To be more specific, the nominator of the fee percentages will be greater than the denominator. Hence, the fee percentages will be invalid.

```
function changeSubscribeFee(uint256 _fee) external onlyAdmin returns (bool) {
    subscribeFee = _fee;
    emit SubscribeFeeUpdated(subscribeFee);
    return true;
}

function changeRedeemFee(uint256 _fee) external onlyAdmin returns (bool) {
    redeemFee = _fee;
    emit SubscribeFeeUpdated(redeemFee);
    return true;
}

function changeFee(uint256 _fee) external onlyAdmin returns (bool) {
    fee = _fee;
    emit FeeUpdated(fee);
    return true;
}


function changeManagerFee(uint256 _managerFee) external onlyAdmin returns (bool){
    managerFee = _managerFee;
    emit ManagerFeeUpdated(managerFee);
    return true;
}
```

```
//AltairFunds
function updateManagerMaxMonthlyFee(uint256 _fee) external {
    onlyFactory();
    managerMaxMonthlyFee = _fee;
}
```

## Recommendation

The contract could embody a check for the maximum acceptable value.

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

# STC - Succeeded Transfer Check

| | |
|---|---|
| **Criticality** | minor / informative |
| **Location** | contract.sol/AltairFactory.sol#L189,191 |
| **Status** | Unresolved |

## Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
naliToken.transferFrom(msg.sender, address(treasury), monthlyCost);

customToken.transferFrom( msg.sender,address(treasury),monthlyCost );
```

## Recommendation

The contract should check if the result of the transfer methods is successful.

# MC - Missing Check

| | |
|---|---|
| **Criticality** | medium |
| **Location** | contract.sol/AltairFactory.sol#L98,203<br>contract.sol/AltairFunds.sol#L154,1278,325<br>contract.sol/AltairMaster.sol#L37 |
| **Status** | Unresolved |

## Description

The contract is processing variables that have not been properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues.

The initializer's arguments are not sanitized properly.

```
function initialize(
    address _proxyCallContract,
    address _rolesContract,
    IAltairShop _shop,
    IAltairSwap _altairSwap,
    ITreasury _treasury,
    IAltairSwapInfo _swapContract,
    address _naliAltair,
    address _naliToken,
    address _naliNftTicket,
    address _customToken,
    IAltairMaster _fundMaster,
    address _NATIVE
)

function changeCustomToken(address _customToken)
    external
    onlyAdmin
    returns (bool)
{
    customToken = IERC20Upgradeable(_customToken);
    return true;
}
```

```
function changeCustomNative(address _native)
    external
    onlyAdmin
    returns (bool)
{
    NATIVE = _native;
    return true;
}
```

The function's arguments are not sanitized properly.

```
function changeCustomToken(address _customToken) external onlyAdmin returns (bool){
    customToken = IERC20Upgradeable(_customToken);
    return true;
}

function changeCustomNative(address _native) external onlyAdmin returns (bool){
    NATIVE = _native;
    return true;
}

//AltairFunds
function initialize(
    string memory _name,
    string memory _symbol,
    address _managerOwner,
    address _factoryOwner,
    bool _community,
    uint256 _version
) public initializer {

//AltairMaster
function initialize(
    address _routerAddress,
    address _priceFeedAddress,
    address _oracle1Inch,
    address _chainLinkPriceFeed,
    address _factory
) public initializer {

function changeRouter(address _address) public onlyAdmin returns (bool) {
    pancakeSwapRouter = IPancakeRouter(_address);
    return true;
}

function _createTargetNames(
    address[] memory _toAddresses,
    uint256[] memory _targetWeight
) internal {
```

```
    //..
    for (uint256 i; i < _toAddresses.length; i++) {
        TargetWeight[_toAddresses[i]] = _targetWeight[i];
        targetNamesAddress.push(_toAddresses[i]);
        //reset nonBM mapping if it is bm target name
        nonBMNamesMapping[_toAddresses[i]] = false;
    }
}

function _handleFeeTransferSubscribe(uint256 _swapOutput)
    internal
    returns (uint256 finalSwapOutput)
{
    platformFee = IAltairFactory(factory).subscribeFee();
     //..
    managerFeeBps = IAltairFactory(factory).managerFee();
    //..
    finalSwapOutput = _swapOutput.sub(platformUnit).sub(managerUnit);
    return (finalSwapOutput);
}
```

## Recommendation

The contract should properly check the variables according to the required specifications.

- The address and the interfaces arguments should not be set to zero address.
- The TargetWeight[targetAdd] should be less than 10000.
- The sum of the platformFee and managerFeeBps should be lower than 10000.

# MTP - Misuse of Transaction Property

| | |
|---|---|
| **Criticality** | minor / informative |
| **Location** | contract.sol/AltairFactory.sol#L141,158<br>contract.sol/AltairFund.sol#704,780 |
| **Status** | Unresolved |

## Description

The contract is processing an argument to illustrate the functions msg.sender.

```
function mint(
    address _fund,
    address _account,
    uint256 _amount,
    bytes memory _data
) external returns (bool) {
    require(msg.sender == _fund, "Sender not fund");
    require(funds.contains(_fund));
    uint256 _id = getFundId(_fund);
    naliAltair.mint(_account, _id, _amount, _data);
    return true;
}
function burn(
    address _fund,
    address _account,
    uint256 _amount
) external returns (bool) {
    require(msg.sender == _fund, "Sender not fund");
    require(funds.contains(_fund));
    bool isApproved = naliAltair.isApprovedForAll(_account, address(this));
    if (!isApproved) {
        naliAltair.setApprovalForTp(_account, address(this), true);
    }
    uint256 _id = getFundId(_fund);
    naliAltair.burn(_account, _id, _amount);
    return true;
}

function subscribe(
    AltairLib.TradeParams memory _tradeParams,
```

```
    address _investorAddress
) external payable nonReentrant returns (uint256) {
    require(pause == false, "Pause");
    require(targetNamesAddress.length > 0, "NoTarget");
    require(
        msg.sender == _investorAddress || msg.sender == factory.shop(),
        "Only Authorized"
    );

function redeem(
    AltairLib.TradeParams memory _tradeParams,
    address _investorAddress
) public payable nonReentrant returns (uint256) {
    require(
        msg.sender == _investorAddress || msg.sender == factory.shop(),
        "Only Authorized"
    );
```

## Recommendation

The contract should only use the transaction property msg.sender.

# L02 - State Variables could be Declared Constant

| | |
|---|---|
| **Criticality** | minor / informative |
| **Location** | contracts/AltairFactory.sol#L43 |
| **Status** | Unresolved |

## Description

Constant state variables should be declared constant to save gas.

```
defaultOperator
```

## Recommendation

Add the constant attribute to state variables that never change.

# L04 - Conformance to Solidity Naming Conventions

| | |
|---|---|
| **Criticality** | minor / informative |
| **Location** | contracts/AltairFactory.sol#L91,106,235,107,223,143,160,159,99,375,136,54,144,104,145,108,269,222,321,180,255,75,423,109,263,224,132,101,305,142,361,453,110,341,105,100,212,281,247,386,349,427,311,203,368,293,385,103,161,102<br><br>contracts/NaliAltair.sol#L24,53,39,61<br><br>contracts/AltairMaster.sol#L232,231,69,39,90,33,105,158,122,138,88,241,38,190,61,40,89,41,42 |
| **Status** | Unresolved |

## Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
redeemFeeUpdated
_naliToken
_fee
_naliNftTicket
_account
_fund
_proxyCallContract
_rolesContract
NATIVE
_useSupportingFeeOnTransferTokens
_tokenaddress
_address
_priceFeedAddress
_usePancake
TokenNames
_from
_to
_toTokenName
```

```
...
```

## Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions.

# L09 - Dead Code Elimination

| | |
|---|---|
| **Criticality** | minor / informative |
| **Location** | contracts/libs/HomoraMath.sol#L24,10,18,14<br>contracts/AltairMaster.sol#L97 |
| **Status** | Unresolved |

## Description

Functions that are not used in the contract, and make the code's size bigger.

```
sqrt
divCeil
fdiv
fmul
_getTokenName
```

## Recommendation

Remove unused functions.

# L11 - Unnecessary Boolean equality

| | |
|---|---|
| **Criticality** | minor / informative |
| **Location** | contracts/AltairFactory.sol#L180,499<br>contracts/AltairMaster.sol#L105,122 |
| **Status** | Unresolved |

## Description

The comparison to boolean constants is redundant. Boolean constants can be used directly and do not need to be compared to true or false.

```
_useNali == true
IAltairFund(fundsArray[i]).copytrading() == true
IAltairFund(fundsArray[i_scope_0]).copytrading() == true


pancakePriceToken[_from] == true
pancakePriceToken[_targetAdd] == true
```

## Recommendation

Remove the equality to the boolean constant.

# L13 - Divide before Multiply Operation

| | |
|---|---|
| **Criticality** | minor / informative |
| **Location** | contracts/AltairMaster.sol#L190 |
| **Status** | Unresolved |

## Description

Performing divisions before multiplications may cause lose of prediction.

```
lpTokenPrice = tokenReserveCumulative.mul(px1).mul(2).div(totalSupply)
```

## Recommendation

The multiplications should be prior to the divisions.

# L14 - Uninitialized Variables in Local Scope

| | |
|---|---|
| **Criticality** | minor / informative |
| **Location** | contracts/AltairFactory.sol#L486,443,467,491,510,462,518<br>contracts/AltairMaster.sol#L92,82,234 |
| **Status** | Unresolved |

## Description

The are variables that are defined in the local scope and are not initialized.

```
i
l_scope_0

i
```

## Recommendation

All the local scoped variables should be initialized.

# Contract Functions

| Contract | Type | Bases | | |
|---|---|---|---|---|
| | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
| | | | | |
| **OwnableUpgradeable** | Implementation | Initializable, ContextUpgradeable | | |
| | __Ownable_init | Internal | ✓ | onlyInitializing |
| | __Ownable_init_unchained | Internal | ✓ | onlyInitializing |
| | owner | Public | | - |
| | _checkOwner | Internal | | |
| | renounceOwnership | Public | ✓ | onlyOwner |
| | transferOwnership | Public | ✓ | onlyOwner |
| | _transferOwnership | Internal | ✓ | |
| | | | | |
| **ClonesUpgradeable** | Library | | | |
| | clone | Internal | ✓ | |
| | cloneDeterministic | Internal | ✓ | |
| | predictDeterministicAddress | Internal | | |
| | predictDeterministicAddress | Internal | | |
| | | | | |
| **Initializable** | Implementation | | | |
| | _disableInitializers | Internal | ✓ | |
| | | | | |
| **PausableUpgradeable** | Implementation | Initializable, ContextUpgradeable | | |
| | __Pausable_init | Internal | ✓ | onlyInitializing |
| | __Pausable_init_unchained | Internal | ✓ | onlyInitializing |
| | paused | Public | | - |
| | _requireNotPaused | Internal | | |
| | _requirePaused | Internal | | |
| | _pause | Internal | ✓ | whenNotPaus |

| | | | | |
|---|---|---|---|---|
| | | | | ed |
| | _unpause | Internal | ✓ | whenPaused |
| | | | | |
| **ERC1155Upgradeable** | Implementation | Initializable, ContextUpgradeable, ERC165Upgradeable, IERC1155Upgradeable, IERC1155MetadataURIUpgradeable | | |
| | __ERC1155_init | Internal | ✓ | onlyInitializing |
| | __ERC1155_init_unchained | Internal | ✓ | onlyInitializing |
| | supportsInterface | Public | | - |
| | uri | Public | | - |
| | balanceOf | Public | | - |
| | balanceOfBatch | Public | | - |
| | setApprovalForAll | Public | ✓ | - |
| | isApprovedForAll | Public | | - |
| | safeTransferFrom | Public | ✓ | - |
| | safeBatchTransferFrom | Public | ✓ | - |
| | _safeTransferFrom | Internal | ✓ | |
| | _safeBatchTransferFrom | Internal | ✓ | |
| | _setURI | Internal | ✓ | |
| | _mint | Internal | ✓ | |
| | _mintBatch | Internal | ✓ | |
| | _burn | Internal | ✓ | |
| | _burnBatch | Internal | ✓ | |
| | _setApprovalForAll | Internal | ✓ | |
| | _beforeTokenTransfer | Internal | ✓ | |
| | _afterTokenTransfer | Internal | ✓ | |
| | _doSafeTransferAcceptanceCheck | Private | ✓ | |
| | _doSafeBatchTransferAcceptanceCheck | Private | ✓ | |
| | _asSingletonArray | Private | | |

| | | | | |
|---|---|---|---|---|
| **ERC1155BurnableUpgradeable** | Implementation | Initializable, ERC1155Upgradeable | | |
| | __ERC1155Burnable_init | Internal | ✓ | onlyInitializing |
| | __ERC1155Burnable_init_unchained | Internal | ✓ | onlyInitializing |
| | burn | Public | ✓ | - |
| | burnBatch | Public | ✓ | - |
| | | | | |
| **ERC1155SupplyUpgradeable** | Implementation | Initializable, ERC1155Upgradeable | | |
| | __ERC1155Supply_init | Internal | ✓ | onlyInitializing |
| | __ERC1155Supply_init_unchained | Internal | ✓ | onlyInitializing |
| | totalSupply | Public | | - |
| | exists | Public | | - |
| | _beforeTokenTransfer | Internal | ✓ | |
| | | | | |
| **ERC1155URIStorageUpgradeable** | Implementation | Initializable, ERC1155Upgradeable | | |
| | __ERC1155URIStorage_init | Internal | ✓ | onlyInitializing |
| | __ERC1155URIStorage_init_unchained | Internal | ✓ | onlyInitializing |
| | uri | Public | | - |
| | _setURI | Internal | ✓ | |
| | _setBaseURI | Internal | ✓ | |
| | | | | |
| **IERC1155MetadataURIUpgradeable** | Interface | IERC1155Upgradeable | | |
| | uri | External | | - |
| | | | | |
| **IERC1155ReceiverUpgradeable** | Interface | IERC165Upgradeable | | |
| | onERC1155Received | External | ✓ | - |
| | onERC1155BatchReceived | External | ✓ | - |
| | | | | |

| | | | | |
|---|---|---|---|---|
| **IERC1155Upgradeable** | Interface | IERC165Upgradeable | | |
| | balanceOf | External | | - |
| | balanceOfBatch | External | | - |
| | setApprovalForAll | External | ✓ | - |
| | isApprovedForAll | External | | - |
| | safeTransferFrom | External | ✓ | - |
| | safeBatchTransferFrom | External | ✓ | - |
| | | | | |
| **ERC1155HolderUpgradeable** | Implementation | Initializable, ERC1155ReceiverUpgradeable | | |
| | __ERC1155Holder_init | Internal | ✓ | onlyInitializing |
| | __ERC1155Holder_init_unchained | Internal | ✓ | onlyInitializing |
| | onERC1155Received | Public | ✓ | - |
| | onERC1155BatchReceived | Public | ✓ | - |
| | | | | |
| **ERC1155ReceiverUpgradeable** | Implementation | Initializable, ERC165Upgradeable, IERC1155ReceiverUpgradeable | | |
| | __ERC1155Receiver_init | Internal | ✓ | onlyInitializing |
| | __ERC1155Receiver_init_unchained | Internal | ✓ | onlyInitializing |
| | supportsInterface | Public | | - |
| | | | | |
| **IERC20PermitUpgradeable** | Interface | | | |
| | permit | External | ✓ | - |
| | nonces | External | | - |
| | DOMAIN_SEPARATOR | External | | - |
| | | | | |
| **IERC20Upgradeable** | Interface | | | |
| | totalSupply | External | | - |
| | balanceOf | External | | - |
| | transfer | External | ✓ | - |

| | | | | |
|---|---|---|---|---|
| | allowance | External | | - |
| | approve | External | ✓ | - |
| | transferFrom | External | ✓ | - |
| | | | | |
| **SafeERC20Upgradeable** | Library | | | |
| | safeTransfer | Internal | ✓ | |
| | safeTransferFrom | Internal | ✓ | |
| | safeApprove | Internal | ✓ | |
| | safeIncreaseAllowance | Internal | ✓ | |
| | safeDecreaseAllowance | Internal | ✓ | |
| | safePermit | Internal | ✓ | |
| | _callOptionalReturn | Private | ✓ | |
| | | | | |
| **AddressUpgradeable** | Library | | | |
| | isContract | Internal | | |
| | sendValue | Internal | ✓ | |
| | functionCall | Internal | ✓ | |
| | functionCall | Internal | ✓ | |
| | functionCallWithValue | Internal | ✓ | |
| | functionCallWithValue | Internal | ✓ | |
| | functionStaticCall | Internal | | |
| | functionStaticCall | Internal | | |
| | verifyCallResult | Internal | | |
| | | | | |
| **ContextUpgradeable** | Implementation | Initializable | | |
| | __Context_init | Internal | ✓ | onlyInitializing |
| | __Context_init_unchained | Internal | ✓ | onlyInitializing |
| | _msgSender | Internal | | |
| | _msgData | Internal | | |
| | | | | |
| **ERC165Upgradeable** | Implementation | Initializable, IERC165Upgradeable | | |

| | | | | |
|---|---|---|---|---|
| | __ERC165_init | Internal | ✓ | onlyInitializing |
| | __ERC165_init_unchained | Internal | ✓ | onlyInitializing |
| | supportsInterface | Public | | - |
| | | | | |
| **IERC165Upgradeable** | Interface | | | |
| | supportsInterface | External | | - |
| | | | | |
| **StringsUpgradeable** | Library | | | |
| | toString | Internal | | |
| | toHexString | Internal | | |
| | toHexString | Internal | | |
| | toHexString | Internal | | |
| | | | | |
| **EnumerableSetUpgradeable** | Library | | | |
| | _add | Private | ✓ | |
| | _remove | Private | ✓ | |
| | _contains | Private | | |
| | _length | Private | | |
| | _at | Private | | |
| | _values | Private | | |
| | add | Internal | ✓ | |
| | remove | Internal | ✓ | |
| | contains | Internal | | |
| | length | Internal | | |
| | at | Internal | | |
| | values | Internal | | |
| | add | Internal | ✓ | |
| | remove | Internal | ✓ | |
| | contains | Internal | | |
| | length | Internal | | |
| | at | Internal | | |
| | values | Internal | | |
| | add | Internal | ✓ | |

| | | | | |
|---|---|---|---|---|
| | remove | Internal | ✓ | |
| | contains | Internal | | |
| | length | Internal | | |
| | at | Internal | | |
| | values | Internal | | |
| | | | | |
| **INaliNftTicket** | Interface | | | |
| | canAccess | External | | - |
| | | | | |
| **AltairFactory** | Implementation | Initializable, ERC1155HolderUpgradeable | | |
| | initialize | Public | ✓ | initializer |
| | getFundId | Public | | - |
| | totalSupply | External | | - |
| | mint | External | ✓ | - |
| | burn | External | ✓ | - |
| | subscribe | External | ✓ | - |
| | changeCustomToken | External | ✓ | onlyAdmin |
| | changeCustomNative | External | ✓ | onlyAdmin |
| | changeNaliTokens | External | ✓ | onlyAdmin |
| | changeSubscribeFee | External | ✓ | onlyAdmin |
| | changeRedeemFee | External | ✓ | onlyAdmin |
| | changeFee | External | ✓ | onlyAdmin |
| | setUseSwapInfo | External | ✓ | onlyAdmin |
| | changeManagerFee | External | ✓ | onlyAdmin |
| | setTreasury | External | ✓ | onlyAdmin |
| | setMaster | External | ✓ | onlyAdmin |
| | setShop | External | ✓ | onlyAdmin |
| | setAltairSwap | External | ✓ | onlyAdmin |
| | setSwapContract | External | ✓ | onlyAdmin |
| | getFee | External | | - |
| | getManagerFee | External | | - |
| | changeMonthlyCost | External | ✓ | onlyAdmin |
| | changeFreeTrial | External | ✓ | onlyAdmin |

| | | | | |
|---|---|---|---|---|
| | adminUpdateImplementation | External | ✓ | onlyAdmin |
| | adminUpdateProxyCallContract | External | ✓ | onlyAdmin |
| | adminUpdateRolesContract | External | ✓ | onlyAdmin |
| | create | External | ✓ | - |
| | fundAt | External | | - |
| | containsFund | External | | - |
| | numberOfFunds | External | | - |
| | getFunds | External | | - |
| | updateAllMaxManagerMonthlyFee | External | ✓ | - |
| | updateAllPlatformAddresses | External | ✓ | - |
| | updateAllCopyTrading | External | ✓ | - |
| | _updateRolesContract | Private | ✓ | |
| | _updateProxyCallContract | Private | ✓ | |
| | _updateImplementation | Private | ✓ | |
| | predictAddress | External | | - |
| | _getSalt | Private | | |
| | onERC1155Received | Public | ✓ | - |
| | onERC1155BatchReceived | Public | ✓ | - |
| | <Receive Ether> | External | Payable | - |
| | | | | |
| **IAltairFund** | Interface | | | |
| | executeAuthorized | External | ✓ | - |
| | setAuthorized | External | ✓ | - |
| | setDCA | External | ✓ | - |
| | updateManagerMaxMonthlyFee | External | ✓ | - |
| | version | External | | - |
| | authorized | External | | - |
| | community | External | | - |
| | dca | External | | - |
| | copytrading | External | | - |
| | sltp | External | | - |
| | executeCustomTx | External | ✓ | - |
| | transferWETH | External | Payable | - |
| | managerMonthlyTimestamp | External | | - |
| | getTargetWeight | External | | - |

| | | | | |
|---|---|---|---|---|
| | managerOwner | External | | - |
| | getTargetWeightQty | External | | - |
| | getBalance | External | | - |
| | totalSupply | External | | - |
| | getUnitPrice | External | | - |
| | getUnitPriceInUSD | External | | - |
| | getFundDataAll | External | | - |
| | getFundValues | External | | - |
| | getNonBMLength | External | | - |
| | updateManagerProperty | External | Payable | - |
| | updateManagerFee | External | Payable | - |
| | updateRebalancePeriod | External | Payable | - |
| | redeem | External | Payable | - |
| | rebalance | External | Payable | - |
| | subscribe | External | Payable | - |
| | moveNonIndexNameToBase | External | ✓ | - |
| | createTargetNames | External | Payable | - |
| | emergencyRedeem | External | Payable | - |
| | getTargetNamesAddress | External | | - |
| | getTargetWeightsAddress | External | | - |
| | updatePlatformAddresses | External | ✓ | - |
| | name | External | | - |
| | fund | External | | - |
| | symbol | External | | - |
| | updateCopyTrading | External | ✓ | - |
| | rebalanceCycle | External | | - |
| | getTransferAmt | External | | - |
| | nonBMNamesMapping | External | | - |
| | nonBMNamesAddress | External | | - |
| | | | | |
| **IAltairMaster** | Interface | | | |
| | useChainlinkOracle | External | | - |
| | getPair | External | | - |
| | getLpPrice | External | | - |
| | getTokenName | External | | - |

| | | | | |
|---|---|---|---|---|
| | getPriceByAddress | External | | - |
| | getPancakePrice | External | | - |
| | getPriceFromBand | External | | - |
| | getRouterAddress | External | | - |
| | getUseSupportingFeeOnTransferTokens | External | | - |
| | | | | |
| **IAltairShop** | Interface | | | |
| | trade | External | ✓ | - |
| | | | | |
| **IAltairSwap** | Interface | | | |
| | swapBNBToTokens | External | Payable | - |
| | swapTokenToBNB | External | Payable | - |
| | addLiquidity | External | Payable | - |
| | removeLiquidity | External | ✓ | - |
| | | | | |
| **IAltairSwapInfo** | Interface | | | |
| | getAmountsOut | External | | - |
| | getAmountsIn | External | | - |
| | cBurgerSwapRouter | External | | - |
| | cBakerySwapRouter | External | | - |
| | cPancakeSwapRouter | External | | - |
| | cMSwapMemoryAddr | External | | - |
| | | | | |
| **IContractInitializer** | Interface | | | |
| | initialize | External | ✓ | - |
| | | | | |
| **INaliToken** | Interface | | | |
| | totalSupply | External | | - |
| | transferFrom | External | ✓ | - |
| | approve | External | ✓ | - |
| | balanceOf | External | | - |
| | internalBalanceOf | External | | - |
| | allowance | External | | - |

| | | | | |
|---|---|---|---|---|
| | | | | |
| **IProxyCall** | Interface | | | |
| | proxyCallAndReturnAddress | External | ✓ | - |
| | | | | |
| **IRoles** | Interface | | | |
| | isAdmin | External | | - |
| | isOperator | External | | - |
| | | | | |
| **ITreasury** | Interface | | | |
| | addLiquidity | External | ✓ | - |
| | addStableLiquidity | External | ✓ | - |
| | addLiquidityFee | External | ✓ | - |
| | buyBack | External | ✓ | - |
| | buyBackFee | External | ✓ | - |
| | devFee | External | ✓ | - |
| | rewardPoolFee | External | ✓ | - |
| | rewardPoolAddress | External | ✓ | - |
| | devAddress | External | ✓ | - |
| | | | | |
| **IWETH** | Interface | | | |
| | deposit | External | Payable | - |
| | transfer | External | ✓ | - |
| | withdraw | External | ✓ | - |
| | | | | |
| **AltairLib** | Library | | | |
| | | | | |
| **LibrarySwapData** | Library | | | |
| | | | | |

| | | | | |
|---|---|---|---|---|
| **NaliAltair** | Implementation | Initializable, ERC1155Upgradeable, OwnableUpgradeable, PausableUpgradeable, ERC1155BurnableUpgradeable, ERC1155SupplyUpgradeable | | |
| | initialize | Public | ✓ | initializer |
| | setRolesContract | External | ✓ | onlyAdmin |
| | uri | Public | | - |
| | setURI | Public | ✓ | onlyAdmin |
| | pause | Public | ✓ | onlyAdmin |
| | unpause | Public | ✓ | onlyAdmin |
| | mint | Public | ✓ | onlyAdmin |
| | mintBatch | Public | ✓ | onlyAdmin |
| | _beforeTokenTransfer | Internal | ✓ | whenNotPaused |
| | setApprovalForTp | Public | ✓ | onlyAdmin |
| | | | | |
| **AltairMaster** | Implementation | Initializable, OwnableUpgradeable | | |
| | initialize | Public | ✓ | initializer |
| | getTokenName | External | | - |
| | changeRouter | Public | ✓ | onlyAdmin |
| | getRouterAddress | External | | - |
| | updateTokenNames | External | ✓ | onlyAdmin |
| | addPancakePriceToken | Public | ✓ | onlyAdmin |
| | _getTokenName | Internal | | |
| | getPriceByAllAddress | External | | - |
| | getPriceByAddress | External | | - |
| | getPriceFromBand | External | | - |
| | getQuotes | Public | | - |
| | getPancakePrice | Public | | - |

| | | | | |
|---|---|---|---|---|
| | sqrt | Public | | - |
| | getLpPrice | External | | - |
| | use1inchOracle | External | | - |
| | useChainlinkOracle | External | | - |
| | updateUseSupportingFeeOnTransferTokens | Public | ✓ | onlyAdmin |
| | getUseSupportingFeeOnTransferTokens | External | | - |
| | getPair | External | | - |
| | | | | |
| **TransferHelper** | Library | | | |
| | safeApprove | Internal | ✓ | |
| | safeTransfer | Internal | ✓ | |
| | safeTransferFrom | Internal | ✓ | |
| | safeTransferBNB | Internal | ✓ | |

# Contract Flow

# Domain Info

| | |
|---|---|
| **Domain Name** | nali.finance |
| **Registry Domain ID** | c3dcca52c09349afbd0f7b38f3f5cf16-DONUTS |
| **Creation Date** | 2021-06-08T07:07:24Z |
| **Updated Date** | 2022-06-06T07:43:05Z |
| **Registry Expiry Date** | 2023-06-08T07:07:24Z |
| **Registrar WHOIS Server** | whois.namecheap.com |
| **Registrar URL** | https://www.namecheap.com/ |
| **Registrar** | NameCheap, Inc. |
| **Registrar IANA ID** | 1068 |

The domain was created over 1 year before the creation of the audit. It will expire in 7 months.

There is no public billing information, the creator is protected by the privacy settings.

# Summary

The Altair Factory ecosystem operates as a funds investment mechanism. This audit focused on investigating possible security issues, business logic concerns, and potential improvements.

We kindly advice the team to re-consider the contract’s implementation since it produces a lot of inaccuracies.

# Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope’s prior written consent. This report is not nor should be considered an “endorsement” or “disapproval” of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope’s position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

# About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io