Cyberscope

Audit Report

Arcadify

July 2023

Network	ETH
Address	0x148654fd31ec3c714c23beacf424a77054f16fa6
Audited by	© cyberscope

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Unresolved
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Unresolved
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	DKO	Delete Keyword Optimization	Unresolved
●	FSA	Fixed Swap Address	Unresolved
●	FRV	Fee Restoration Vulnerability	Unresolved
●	MEE	Missing Events Emission	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L07	Missing Events Arithmetic	Unresolved

Table of Contents

Review

Explorer	https://etherscan.io/address/0x148654fd31ec3c714c23beacf424a77054f16fa6

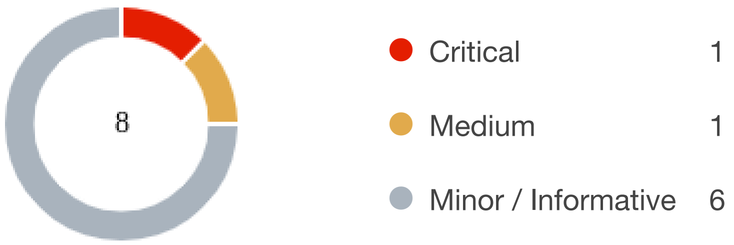
Audit Updates

Initial Audit	12 Jul 2023 https://github.com/cyberscope-io/audits/blob/main/7-arc/v1/audit.pdf
Corrected Phase 2	14 Jul 2023 https://github.com/cyberscope-io/audits/blob/main/7-arc/v2/audit.pdf
Corrected Phase 3	17 Jul 2023 https://github.com/cyberscope-io/audits/blob/main/7-arc/v3/audit.pdf
Corrected Phase 4	19 Jul 2023

Source Files

Filename	SHA256
contracts/arcadify.sol	9e10e00f7a296f15a83266e72d0ae8f9a64b15103adad9e12ffab3ae65733ee7

Findings Breakdown



Severity		Unresolved	Acknowledged	Resolved	Other
●	Critical	1	0	0	0
●	Medium	1	0	0	0
●	Minor / Informative	6	0	0	0

ST - Stops Transactions

Criticality	Critical
Location	contracts/arcadify.sol#L291
Status	Unresolved

Description

The transactions are initially disabled for all users excluding the authorized addresses. The owner can enable the transactions for all users. Once the transactions are enable the owner will not be able to disable them again.

```
if (!tradingOpen) {
    require(
      from == owner(),
      "TOKEN: This account cannot send tokens until trading is enabled"
      );
}
```

Recommendation

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. Some suggestions are:

- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

ELFM - Exceeds Fees Limit

Criticality	Medium
Location	contracts/arcadify.sol#L512
Status	Unresolved

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `setFee` function and set both the `_redisFeeOnBuy` and `_taxFeeOnBuy` or `_redisFeeOnSel` and `_taxFeeOnSelll` to the value of 25. As a result, the total charge will equal 50, since the fees are added and then subtracted from the transferred amount. `(tTransferAmount = tAmount - tFee - tTeam;)`

```
    function setFee(
        uint256 redisFeeOnBuy,
        uint256 redisFeeOnSell,
        uint256 taxFeeOnBuy,
        uint256 taxFeeOnSell
    ) public onlyOwner {
        require(_taxFeeOnBuy <= 25 && _taxFeeOnSell <= 25 &&
redisFeeOnBuy <= 25 && redisFeeOnSell <= 25, "Tax can't be higher than
25");
        _redisFeeOnBuy = redisFeeOnBuy;
        _redisFeeOnSell = redisFeeOnSell;

        _taxFeeOnBuy = taxFeeOnBuy;
        _taxFeeOnSell = taxFeeOnSell;
    }

    function _getTValues(
        uint256 tAmount,
        uint256 redisFee,
        uint256 taxFee
    ) private pure returns (uint256, uint256, uint256) {
        uint256 tFee = tAmount * redisFee / 100;
        uint256 tTeam = tAmount * taxFee / 100;
        uint256 tTransferAmount = tAmount - tFee - tTeam;

        return (tTransferAmount, tFee, tTeam);
     }
```

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. Some suggestions are:

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.
- Renouncing the ownership will eliminate the threats but it is non-reversible.

DKO - Delete Keyword Optimization

Criticality	Minor / Informative
Location	contracts/arcadify.sol#L262
Status	Unresolved

Description

The contract resets variables to the default state by setting the initial values. Setting values to state variables increases the gas cost.

```
function removeAllFee() private {
    if (_redisFee == 0 && _taxFee == 0) return;

    _previousredisFee = _redisFee;
    _previoustaxFee = _taxFee;

    _redisFee = 0;
    _taxFee = 0;
}
```

Recommendation

The team is advised to use the `delete` keyword instead of setting variables. This can be more efficient than setting the variable to a new value, using delete can reduce the gas cost associated with storing data on the blockchain.

FSA - Fixed Swap Address

Criticality	Minor / Informative
Location	contracts/arcadify.sol#L179
Status	Unresolved

Description

The swap address is assigned once and it can not be changed. It is a common practice in decentralized exchanges to create new swap versions. A contract that cannot change the swap address may not be able to catch up to the upgrade. As a result, the contract will not be able to migrate to a new liquidity pool pair or decentralized exchange.

```
constructor() {
    ...
    IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(
        0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D // uniswap
    );
    uniswapV2Router = _uniswapV2Router;
    uniswapV2Pair =
IUniswapV2Factory(_uniswapV2Router.factory())
        .createPair(address(this), _uniswapV2Router.WETH());
    ...
}
```

Recommendation

The team is advised to add the ability to change the pair and router address in order to cover potential liquidity pool migrations. It would be better to support multiple pair addresses so the token will be able to have the same behavior in all the decentralized liquidity pairs.

FRV - Fee Restoration Vulnerability

Criticality	Minor / Informative
Location	contracts/arcadify.sol#L262,272,405
Status	Unresolved

Description

The contract demonstrates a potential vulnerability upon removing and restoring the fees. This vulnerability can occur when the fees have been set to zero. During a transaction, if the fees have been set to zero, then both remove fees and restore fees functions will be executed. The remove fees function is executed to temporarily remove the fees, ensuring the sender is not taxed during the transfer. However, the function prematurely returns without setting the variables that hold the previous fee values.

As a result, when the subsequent restore fees function is called after the transfer, it restores the fees to their previous values. However, since the previous fee values were not properly set to zero, there is a risk that the fees will retain their non-zero values from before the fees were removed. This can lead to unintended consequences, potentially causing incorrect fee calculations or unexpected behavior within the contract.

```
/**
 * The given code segment is just an example of this
vulnerability, where the
 * `removeAllFee()` function does not set the previous fee
variables if both
 * `_redisFee` and `_taxFee` are zero. Then, the
`restoreAllFee()` function
 * will modify the `_redisFee` and `_taxFee` to the values the
previous fees
 * hold, which most likely will not be zero.
 */

function removeAllFee() private {
    if (_redisFee == 0 && _taxFee == 0) return;

    _previousredisFee = _redisFee;
    _previoustaxFee = _taxFee;

    _redisFee = 0;
    _taxFee = 0;
}

function restoreAllFee() private {
    _redisFee = _previousredisFee;
    _taxFee = _previoustaxFee;
}

function _tokenTransfer(
    address sender,
    address recipient,
    uint256 amount,
    bool takeFee
) private {
    if (!takeFee) removeAllFee();
    _transferStandard(sender, recipient, amount);
    if (!takeFee) restoreAllFee();
}
...
```

Recommendation

The team is advised to modify the remove fees function to ensure that the previous fee values are correctly set to zero, regardless of their initial values. A recommended approach would be to remove the early return when both fees are zero.

MEE - Missing Events Emission

Criticality	Minor / Informative
Location	contracts/arcadify.sol#L533,545,554
Status	Unresolved

Description

The contract performs actions and state mutations from external methods that do not result in the emission of events. Emitting events for significant actions is important as it allows external parties, such as wallets or dApps, to track and monitor the activity on the contract. Without these events, it may be difficult for external parties to accurately determine the current state of the contract.

```
   function setMinSwapTokensThreshold(
        uint256 swapTokensAtAmount
    ) public onlyOwner {
        _swapTokensAtAmount = swapTokensAtAmount;
    }

    function setMaxTxnAmount(uint256 maxTxAmount) public
onlyOwner {
        require(_maxTxAmount > 0, "Max TX Amount needs to be
larger than 0");
        _maxTxAmount = maxTxAmount;
    }

    function setMaxWalletSize(uint256 maxWalletSize) public
onlyOwner {
        require(_maxWalletSize > 0, "Max wallet size needs to
be larger than 0");
        _maxWalletSize = maxWalletSize;
    }
```

Recommendation

It is recommended to include events in the code that are triggered each time a significant action is taking place within the contract. These events should include relevant details such as the user's address and the nature of the action taken. By doing so, the contract will be more transparent and easily auditable by external parties. It will also help prevent potential issues or disputes that may arise in the future.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	contracts/arcadify.sol#L107,124,125,126,132,139,147,154,156,167,168,169,537
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
function WETH() external pure returns (address);
string private constant _name = "Arcadify"
string private constant _symbol = "ARC"
uint8 private constant _decimals = 8
uint256 private constant _tTotal = 100000000 * 10 ** 8
uint256 public _taxFeeOnBuy = 5
uint256 public _taxFee = _taxFeeOnSell
...
        payable(0x5B10C241ee81fb51241fb7d80586614bd021125B)

address payable private constant _marketingAddress =
        payable(0x9F67D213eCF40D4d339e105F71D6cA47c287FCb6)
uint256 public _maxTxAmount = 100000000 * 10 ** 8
uint256 public _maxWalletSize = 100000000 * 10 ** 8
uint256 public _swapTokensAtAmount = 50000 * 10 ** 8

...
```

Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.
Find more information on the Solidity documentation
https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention.

L07 - Missing Events Arithmetic

Criticality	Minor / Informative
Location	contracts/arcadify.sol#L522,533,544,550
Status	Unresolved

Description

Events are a way to record and log information about changes or actions that occur within a contract. They are often used to notify external parties or clients about events that have occurred within the contract, such as the transfer of tokens or the completion of a task.

It's important to carefully design and implement the events in a contract, and to ensure that all required events are included. It's also a good idea to test the contract to ensure that all events are being properly triggered and logged.

```
_redisFeeOnBuy = redisFeeOnBuy
_swapTokensAtAmount = swapTokensAtAmount
_maxTxAmount = maxTxAmount
_maxWalletSize = maxWalletSize
```

Recommendation

By including all required events in the contract and thoroughly testing the contract's functionality, the contract ensures that it performs as intended and does not have any missing events that could cause issues with its arithmetic.

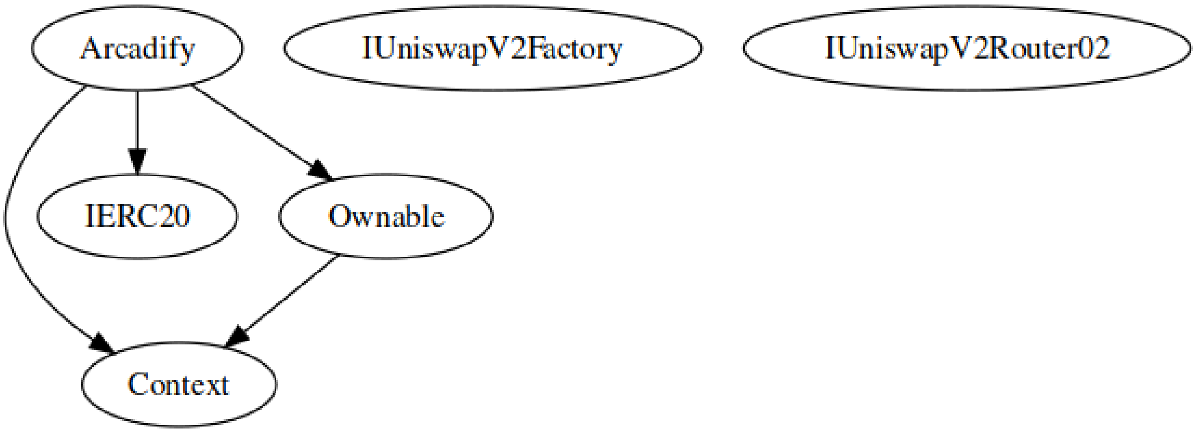
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
	_msgSender	Internal		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
IUniswapV2Factory	Interface			

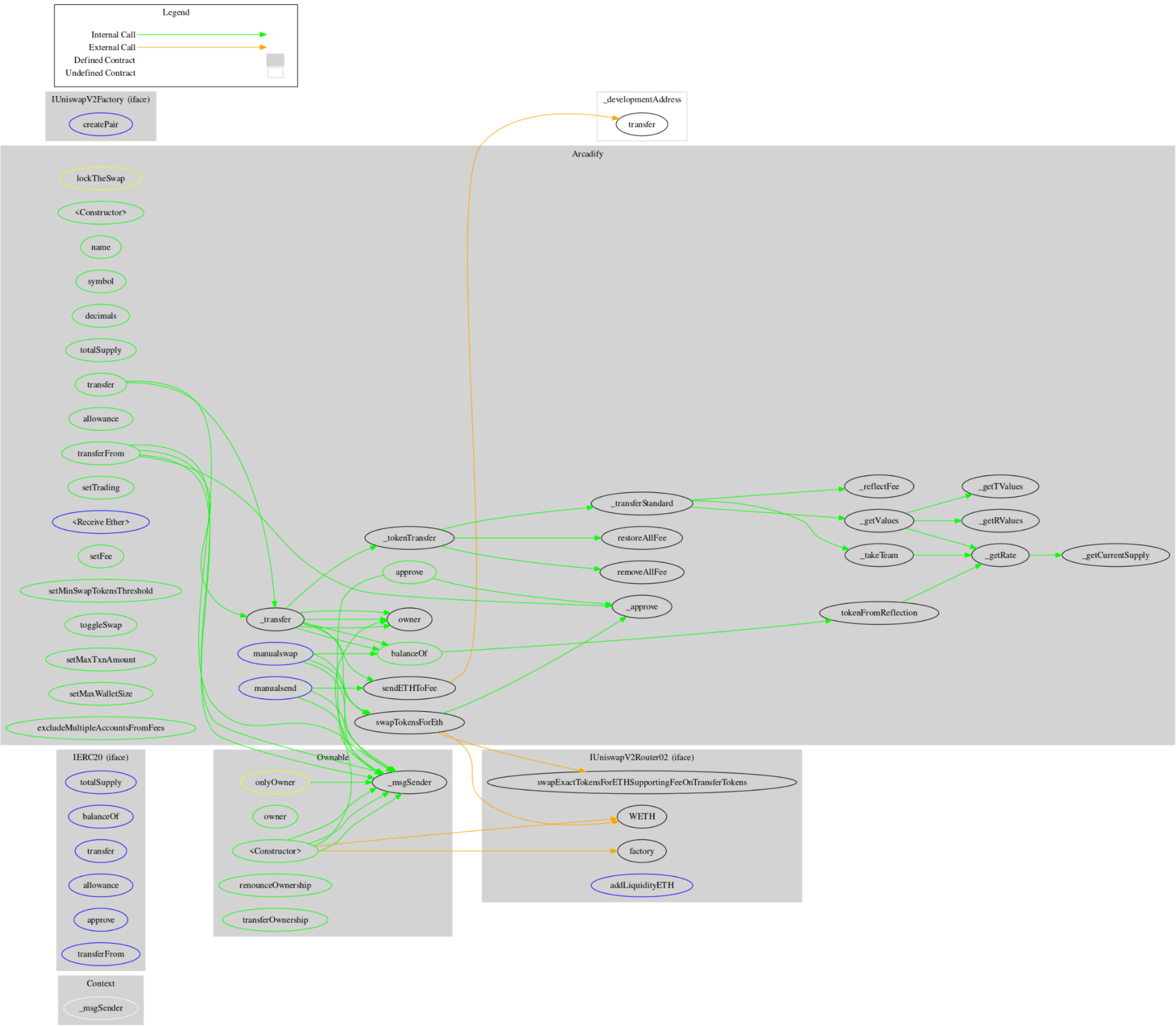
	createPair	External	✓	-
IUniswapV2Router02	Interface			
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
Arcadify	Implementation	Context, IERC20, Ownable		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	tokenFromReflection	Private		
	removeAllFee	Private	✓	
	restoreAllFee	Private	✓	

	_approve	Private	✓	
	_transfer	Private	✓	
	swapTokensForEth	Private	✓	lockTheSwap
	sendETHToFee	Private	✓	
	setTrading	Public	✓	onlyOwner
	manualswap	External	✓	-
	manualsend	External	✓	-
	_tokenTransfer	Private	✓	
	_transferStandard	Private	✓	
	_takeTeam	Private	✓	
	_reflectFee	Private	✓	
		External	Payable	-
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	setFee	Public	✓	onlyOwner
	setMinSwapTokensThreshold	Public	✓	onlyOwner
	toggleSwap	Public	✓	onlyOwner
	setMaxTxnAmount	Public	✓	onlyOwner
	setMaxWalletSize	Public	✓	onlyOwner
	excludeMultipleAccountsFromFees	Public	✓	onlyOwner

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IUniswapV2Factory (iface)
createPair
_developmentAddress
transfer
Arcadify
lockTheSwap
<Constructor>
name
symbol
decimals
totalSupply
transfer
allowance
transferFrom
setTrading
<Receive Ether>
setFee
setMinSwapTokensThreshold
toggleSwap
setMaxTxnAmount
setMaxWalletSize
excludeMultipleAccountsFromFees
_transfer
manualswap
manualsend
_tokenTransfer
approve
owner
balanceOf
sendETHToFee
swapTokensForEth
_transferStandard
restoreAllFee
removeAllFee
_approve
_reflectFee
_getValues
_takeTeam
tokenFromReflection
_getTValues
_getRValues
_getRate
_getCurrentSupply
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Context
_msgSender
Ownable
onlyOwner
owner
<Constructor>
renounceOwnership
transferOwnership
_msgSender
IUniswapV2Router02 (iface)
swapExactTokensForETHSupportingFeeOnTransferTokens
WETH
factory
addLiquidityETH

Summary

Arcadify contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. There are some functions that can be abused by the owner like stop transactions and manipulate the fees. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract will eliminate all the contract threats. There is also a limit of max 50% fee.

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