



Audit Report

A51 Finance

December 2023

Repository <https://github.com/a51finance/A51-token>

Commit `dc2087a2ae2a569b817145219db718a7eab92bd0`

Audited by © cyberscope

Table of Contents

Table of Contents	1
Review	2
Audit Updates	2
Source Files	2
Findings Breakdown	3
Diagnostics	4
ALM - Array Length Mismatch	5
Description	5
Recommendation	5
L17 - Usage of Solidity Assembly	7
Description	7
Recommendation	7
Functions Analysis	8
Flow Graph	9
Summary	10
Disclaimer	11
About Cyberscope	12

Review

Repository	https://github.com/a51finance/A51-token
Commit	dc2087a2ae2a569b817145219db718a7eab92bd0

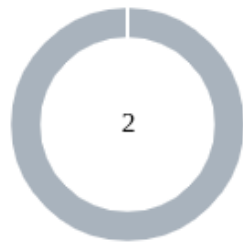
Audit Updates

Initial Audit	27 Dec 2023
---------------	-------------

Source Files

Filename	SHA256
A51.sol	665e11823c465cbc71e3c7bc7577465aa1efbcc5dd7b9bea102b8216af4ccf7d

Findings Breakdown



● Critical	0
● Medium	0
● Minor / Informative	2

Severity	Unresolved	Acknowledged	Resolved	Other
● Critical	0	0	0	0
● Medium	0	0	0	0
● Minor / Informative	2	0	0	0

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	ALM	Array Length Mismatch	Unresolved
●	L17	Usage of Solidity Assembly	Unresolved

ALM - Array Length Mismatch

Criticality	Minor / Informative
Location	A51.sol#L39
Status	Unresolved

Description

The contract is currently utilizing the `_mintToWallets` function to mint tokens to a list of accounts. This function is called within the constructor and takes the `accounts` and `amounts` arrays as input. However, there is no verification to ensure that the lengths of the `accounts` and `amounts` arrays are equal. This oversight can lead to potential issues. If the `accounts` array is longer than the `amounts` array, the contract will attempt to access an index in the `amounts` array that does not exist, resulting in a runtime error and transaction failure. Conversely, if the `amounts` array is longer, some amounts will not be minted to any account, leading to an inconsistency in the intended token distribution.

```
constructor(address[] memory accounts, uint256[] memory amounts)
ERC20("A51 Finance", "A51") {
    _mintToWallets(accounts, amounts);
}

function _mintToWallets(address[] memory _accounts, uint256[] memory
_amounts) internal {
    for (uint256 i = 0; i < _accounts.length; i++) {
        _mint(_accounts[i], _amounts[i]);
    }
}
```

Recommendation

It is recommended to add an additional check in the `_mintToWallets` function to ensure that the lengths of both the `accounts` and `amounts` arrays are the same. This can be implemented as a `require` statement at the beginning of the function, comparing the lengths of the two arrays and reverting the transaction if they do not match. This check will prevent the function from executing if the arrays are not of equal length, thereby avoiding potential runtime errors and ensuring that each account in the `accounts` array has a

corresponding amount in the amounts array. This modification will enhance the robustness and reliability of the token minting process in the contract.

L17 - Usage of Solidity Assembly

Criticality	Minor / Informative
Location	A51.sol#L85
Status	Unresolved

Description

Using assembly can be useful for optimizing code, but it can also be error-prone. It's important to carefully test and debug assembly code to ensure that it is correct and does not contain any errors.

Some common types of errors that can occur when using assembly in Solidity include Syntax, Type, Out-of-bounds, Stack, and Revert.

```
assembly {  
    chainId := chainid()  
}
```

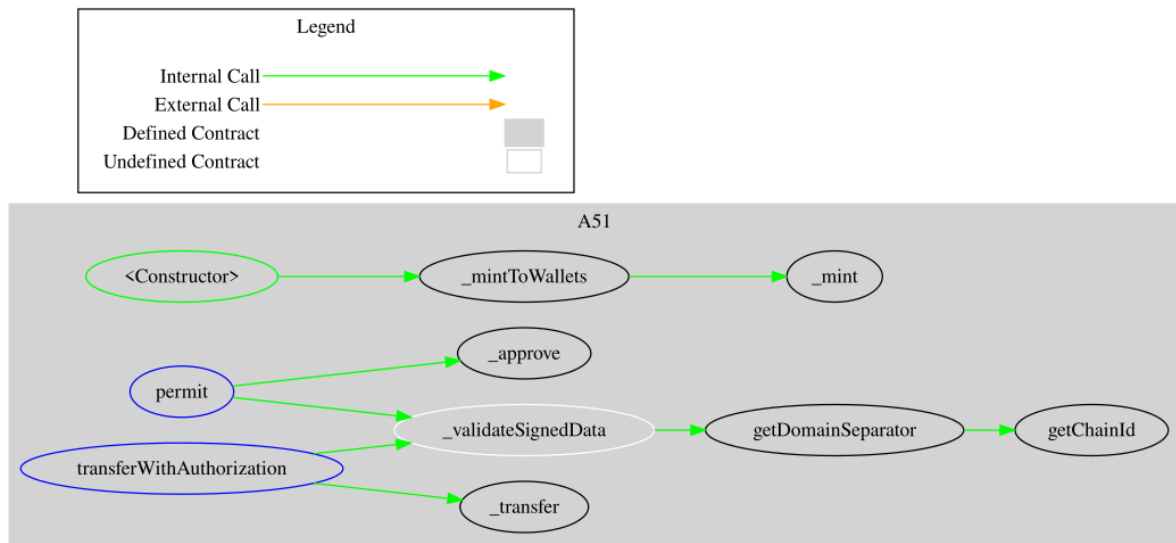
Recommendation

It is recommended to use assembly sparingly and only when necessary, as it can be difficult to read and understand compared to Solidity code.

Functions Analysis

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
A51	Implementation	ERC20Burnable		
		Public	✓	ERC20
	_mintToWallets	Internal	✓	
	_validateSignedData	Internal		
	permit	External	✓	-
	getDomainSeparator	Public		-
	getChainId	Public		-
	transferWithAuthorization	External	✓	-

Flow Graph



Summary

A51 Finance contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. A51 Finance is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions.

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>