Cyberscope

Audit Report

KMM39289793V83739L0KM

November 2024

Network ETH

Address 0xd6E911985f2FD0C57E6A5a4d0595C765679E3195

Audited by © cyberscope

Analysis

Critical Medium Minor / Informative Pass

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

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	LPMDTU	Lack of Pegging Mechanish DWT To USDT	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved
	L19	Stable Compiler Version	Unresolved

Table of Contents

Risk Classification

The criticality of findings in Cyberscope's smart contract audits is determined by evaluating multiple variables. The two primary variables are:

1. **Likelihood of Exploitation**: This considers how easily an attack can be executed, including the economic feasibility for an attacker.
2. **Impact of Exploitation**: This assesses the potential consequences of an attack, particularly in terms of the loss of funds or disruption to the contract's functionality.

Based on these variables, findings are categorized into the following severity levels:

1. **Critical**: Indicates a vulnerability that is both highly likely to be exploited and can result in significant fund loss or severe disruption. Immediate action is required to address these issues.
2. **Medium**: Refers to vulnerabilities that are either less likely to be exploited or would have a moderate impact if exploited. These issues should be addressed in due course to ensure overall contract security.
3. **Minor**: Involves vulnerabilities that are unlikely to be exploited and would have a minor impact. These findings should still be considered for resolution to maintain best practices in security.
4. **Informative**: Points out potential improvements or informational notes that do not pose an immediate risk. Addressing these can enhance the overall quality and robustness of the contract.

Severity	Likelihood / Impact of Exploitation
Critical	Highly Likely / High Impact
Medium	Less Likely / High Impact or Highly Likely/ Lower Impact
Minor / Informative	Unlikely / Low to no Impact

Review

Contract Name	Token
Compiler Version	v0.8.27+commit.40a35a09
Optimization	200 runs
Explorer	https://etherscan.io/address/0xd6e911985f2fd0c57e6a5a4d0595c765679e3195
Address	0xd6e911985f2fd0c57e6a5a4d0595c765679e3195
Network	ETH
Symbol	DWT
Decimals	18
Total Supply	150,000,000
Badge Eligibility	Yes

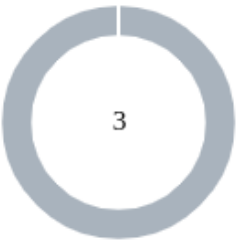
Audit Updates

Initial Audit	15 Nov 2024

Source Files

Filename	SHA256
Token.sol	a1ba19a74be8c5ce9d81ecf743da7eedfae09376ee45f9db75e7c14018099ac8

Findings Breakdown



Critical	0
Medium	0
Minor / Informative	3

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	0	0	0
Minor / Informative	3	0	0	0

LPMDTU - Lack of Pegging Mechanish DWT To USDT

Criticality	Minor / Informative
Location	Token.sol
Status	Unresolved

Description

The `Token` contract is found to have the name and symbol referencing a bank transfer, suggesting the tokens represent fiat-backed assets. The contract initializes with a fixed supply of **150,000,000 DWT tokens**, minted to the address `0x16E04e89D4040CF074DFaf2769A1283f533F7970`.

However, the contract does not include any functionality to facilitate or enforce the stated goal of maintaining a peg where **1 DWT = 1 USDT**. In its current form, the contract simply acts as a standard ERC-20 token without mechanisms to:

1. Manage the relationship between `DWT` and USDT.
2. Enable minting or redeeming of tokens based on corresponding USDT deposits or withdrawals.
3. Ensure collateralization of the circulating token supply with actual USDT reserves.

Recommendation

The team is advised to clearly define and implement mechanisms aligned with their stated goal of pegging **1 DWT = 1 USDT**. To achieve this, the following steps are recommended:

1. **Implement Minting and Redemption Functions:**
    - Add functionality allowing users to mint `DWT` tokens by depositing an equivalent amount of USDT to a designated wallet.
    - Provide a redemption mechanism for users to burn `DWT` tokens in exchange for the corresponding amount of USDT.
2. **Optionally Demonstrate USDT Reserves:**
    - To build user trust and demonstrate that the peg truly exists, consider making the USDT reserves publicly viewable or verified through periodic

third-party audits. This optional step would provide transparency and validate the token's collateralization.

3. **Avoid Arbitrary Token Transfers:**
    - If the tokens are meant to represent 1:1 backed assets, avoid uncontrolled transfers that could break the peg. Instead, consider limiting token minting and burning to only authorized participants or through verified off-chain processes.
4. **Optional Decentralized Pegging Mechanism (if required):**
    - If `DWT` is to be traded on decentralized exchanges, implement price stabilization mechanisms (e.g., dynamic minting/burning or oracle-based price tracking). This ensures that market fluctuations do not compromise the peg.

By integrating these mechanisms, the team can align the contract's functionality with their stated goal, ensuring a 1:1 parity between `DWT` and USDT, and bolstering trust in the token's intended use case.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	Token.sol#L9
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.

```
uint256 constant initialSupply = 150000000 * (10 ** 18)
```

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/stable/style-guide.html#naming-conventions.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	Token.sol#L3
Status	Unresolved

Description

The `^` symbol indicates that any version of Solidity that is compatible with the specified version (i.e., any version that is a higher minor or patch version) can be used to compile the contract. The version lock is a mechanism that allows the author to specify a minimum version of the Solidity compiler that must be used to compile the contract code. This is useful because it ensures that the contract will be compiled using a version of the compiler that is known to be compatible with the code.

```
pragma solidity ^0.8.0;
```

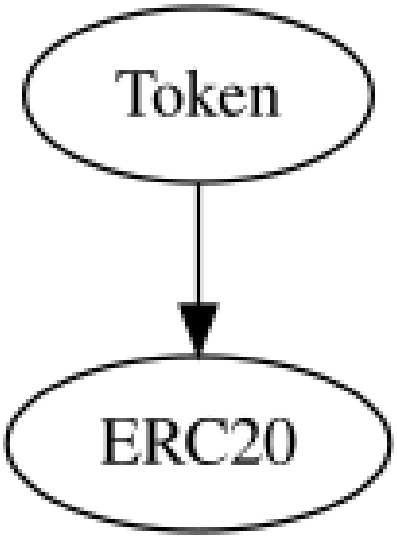
Recommendation

The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest version that provides all the required functionality for the codebase. As a result, the project will be compiled in a well-tested LTS (Long Term Support) environment.

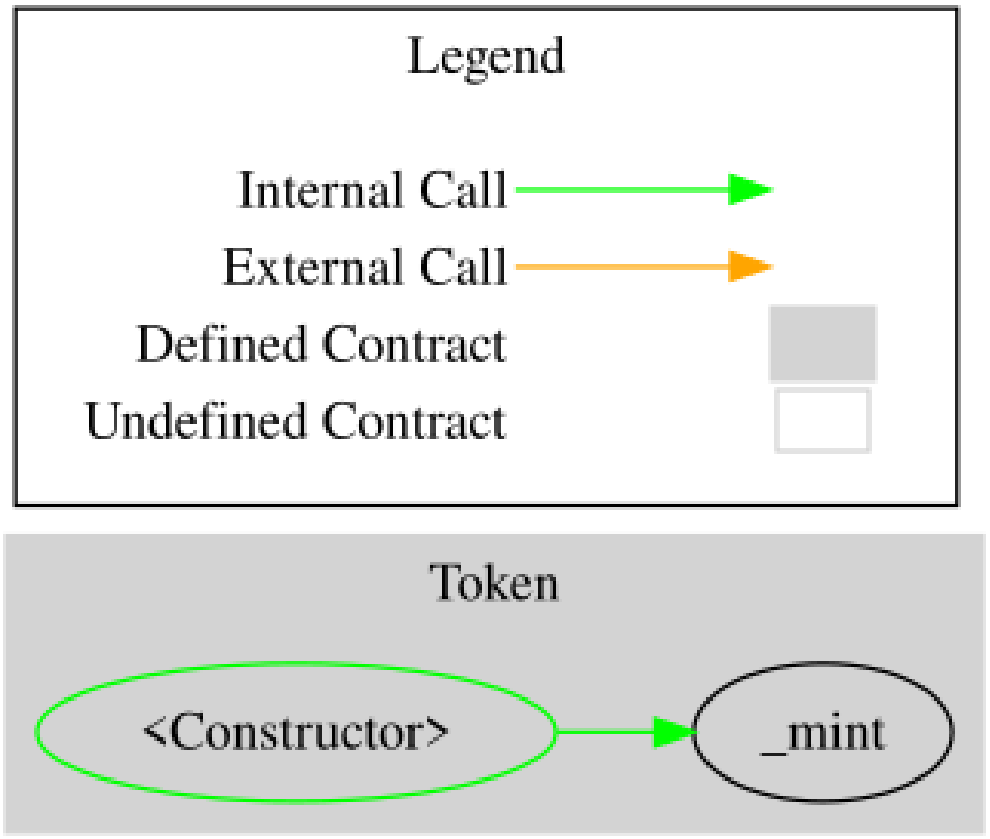
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Token	Implementation	ERC20		
		Public	✓	ERC20

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Token
<Constructor>
_mint

Summary

KMM39289793V83739L0KM contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. KMM39289793V83739L0KM 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

cyberscope.io