



Cyberscope

Audit Report

Tectum Emission Token

May 2023

Network ETH

Address 0x68A47Fe1CF42eBa4a030a10CD4D6a1031Ca3CA0a

Audited by © cyberscope

Table of Contents

Table of Contents	1
Review	2
Audit Updates	3
Source Files	3
Findings Breakdown	4
Analysis	5
Diagnostics	6
L04 - Conformance to Solidity Naming Conventions	7
Description	7
Recommendation	8
Functions Analysis	9
Inheritance Graph	11
Flow Graph	12
Summary	13
Disclaimer	14
About Cyberscope	15

Review

Contract Name	Tettoken
Compiler Version	v0.4.17+commit.bdeb9e52
Optimization	200 runs
Explorer	https://etherscan.io/address/0x68a47fe1cf42eba4a030a10cd4d6a1031ca3ca0a
Address	0x68a47fe1cf42eba4a030a10cd4d6a1031ca3ca0a
Network	ETH
Symbol	TET
Decimals	8
Total Supply	10,000,000

Audit Updates

Initial Audit	09 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v1/audit.pdf
Corrected Phase 2	14 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v2/audit.pdf
Corrected Phase 3	17 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v3/audit.pdf
Corrected Phase 4	18 Apr 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v4/audit.pdf
Corrected Phase 5	05 May 2023

Source Files

Filename	SHA256
TetToken.sol	e8c2c63a22abfd88d8b19abfc9118ba2a50f555f54d0eed5cb86d7347319b0a1

Findings Breakdown



● Critical	0
● Medium	0
● Minor / Informative	1

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

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	L04	Conformance to Solidity Naming Conventions	Unresolved

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	Tettoken.sol#L40,57,97,110,123,131,149,168,212,232,248
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.

```
address _owner
uint256 public _totalSupply
uint256 _value
address _to
uint256 public constant maxtet =1000000000000000
address _from
address _spender
uint256 _newts
uint256 _amount
```

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

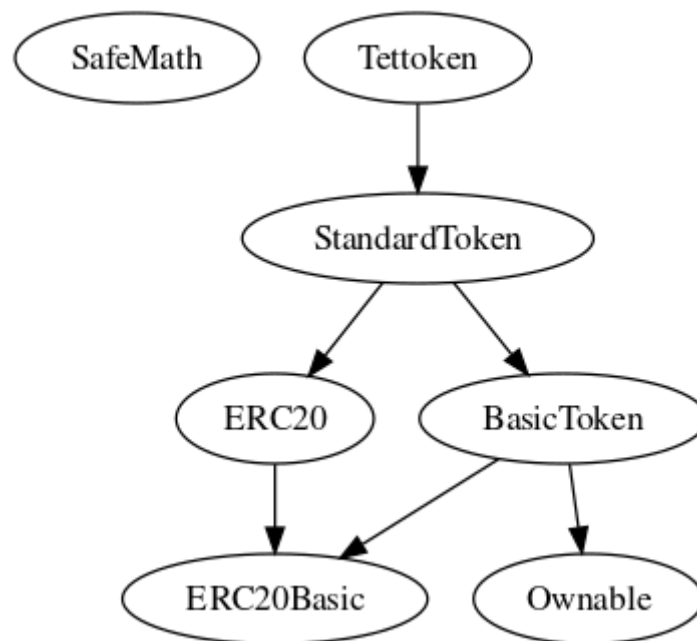
<https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention>.

Functions Analysis

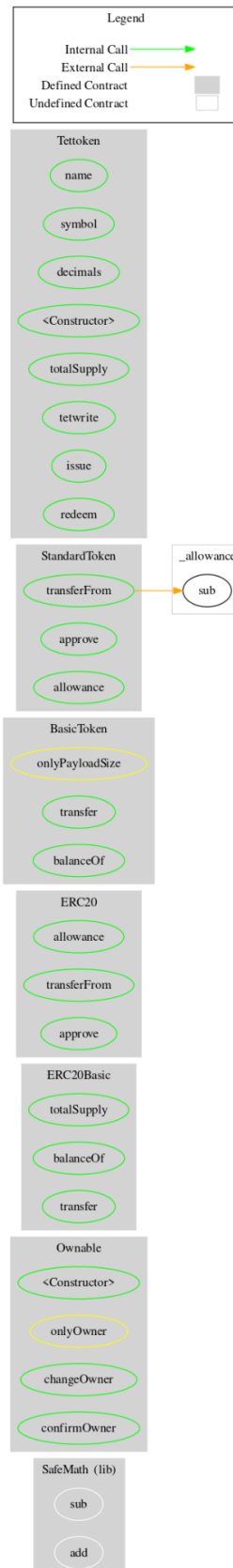
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
SafeMath	Library			
	sub	Internal		
	add	Internal		
Ownable	Implementation			
		Public	✓	-
	changeOwner	Public	✓	onlyOwner
	confirmOwner	Public	✓	-
ERC20Basic	Implementation			
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
ERC20	Implementation	ERC20Basic		
	allowance	Public		-
	transferFrom	Public	✓	-
	approve	Public	✓	-

BasicToken	Implementation	Ownable, ERC20Basic		
	transfer	Public	✓	onlyPayloadSize
	balanceOf	Public		-
StandardToken	Implementation	BasicToken, ERC20		
	transferFrom	Public	✓	onlyPayloadSize
	approve	Public	✓	onlyPayloadSize
	allowance	Public		-
Tettoken	Implementation	StandardToken		
	name	Public		-
	symbol	Public		-
	decimals	Public		-
		Public	✓	-
	totalSupply	Public		-
	tetwrite	Public	✓	onlyOwner
	issue	Public	✓	onlyOwner
	redeem	Public	✓	onlyOwner

Inheritance Graph



Flow Graph



Summary

Tectum Emission Token contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. Tectum Emission Token is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler errors 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>



Cyberscope

Audit Report

Tectum Enumeration Token

March 2023

SHA256 525f930c7764a8c4dd1af1dc116c221604ca16489a739ab95b8ac57e1b916b06

Audited by © cyberscope

Table of Contents

Table of Contents	1
Review	2
Audit Updates	2
Source Files	2
Analysis	3
MT - Mints Tokens	4
Description	4
Recommendation	4
BT - Burns Tokens	6
Description	6
Recommendation	6
Diagnostics	8
CUOO - Calculations Underflow Or Overflow	9
Description	9
Recommendation	9
RII - Redundant Inheritance Issue	10
Description	10
Recommendation	10
L04 - Conformance to Solidity Naming Conventions	11
Description	11
Recommendation	12
L08 - Tautology or Contradiction	13
Description	13
Recommendation	13
L19 - Stable Compiler Version	14
Description	14
Recommendation	14
L23 - ERC20 Interface Misuse	15
Description	15
Recommendation	15
Functions Analysis	16
Inheritance Graph	18
Flow Graph	19
Summary	20
Disclaimer	21
About Cyberscope	22

Review

Audit Updates

Initial Audit	09 Mar 2023
---------------	-------------

Source Files

Filename	SHA256
TETtoken5c_7-3-23.sol	525f930c7764a8c4dd1af1dc116c221604 ca16489a739ab95b8ac57e1b916b06

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Unresolved
●	BT	Burns Tokens	Unresolved
●	BC	Blacklists Addresses	Passed

MT - Mints Tokens

Criticality	Critical
Location	TETtoken5c_7-3-23.sol#L289,311
Status	Unresolved

Description

The contract owner has the authority to mint tokens. The owner may take advantage of it by calling the `TETwrite` or `issue` function. As a result, the contract tokens will be highly inflated.

```
function TETwrite(uint newTS) public onlyOwner returns (uint) {
    require( newTS <= MAX_TET );
    require( newTS != _totalSupply );

    if (newTS > _totalSupply) {
        balances[owner] += ( newTS - _totalSupply );
        _totalSupply = newTS ;
    } else {
        require ( (balances[owner] - ( _totalSupply - newTS )) >= 0 );
        balances[owner] -= ( _totalSupply - newTS );
        _totalSupply = newTS ;
    }

    return _totalSupply ;
    TETwriteEvent( _totalSupply );
}

function issue(uint amount) public onlyOwner {
    require( _totalSupply + amount <= MAX_TET );
    require( _totalSupply + amount > _totalSupply );
    require( balances[owner] + amount > balances[owner] );

    balances[owner] += amount;
    _totalSupply += amount;
    Issue( amount );
}
```

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 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.

BT - Burns Tokens

Criticality	Critical
Location	TETtoken5c_7-3-23.sol#L289,328
Status	Unresolved

Description

The contract owner has the authority to burn tokens from a specific address. The owner may take advantage of it by calling the `TETwrite` or `redeem` function. As a result, the targeted address will lose the corresponding tokens.

```
function TETwrite(uint newTS) public onlyOwner returns (uint) {
    require( newTS <= MAX_TET );
    require( newTS != _totalSupply );

    if (newTS > _totalSupply) {
        balances[owner] += ( newTS - _totalSupply );
        _totalSupply = newTS ;
    } else {
        require ( (balances[owner] - ( _totalSupply - newTS )) >= 0 );
        balances[owner] -= ( _totalSupply - newTS );
        _totalSupply = newTS ;
    }

    return _totalSupply ;
    TETwriteEvent( _totalSupply );
}

function redeem(uint amount) public onlyOwner {
    require(_totalSupply >= amount);
    require(balances[owner] >= amount);

    _totalSupply -= amount;
    balances[owner] -= amount;
    Redeem(amount);
}
```

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 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.

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	CUOO	Calculations Underflow Or Overflow	Unresolved
●	RII	Redundant Inheritance Issue	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L08	Tautology or Contradiction	Unresolved
●	L19	Stable Compiler Version	Unresolved
●	L23	ERC20 Interface Misuse	Unresolved

CUOO - Calculations Underflow Or Overflow

Criticality	Critical
Location	TETtoken5c_7-3-23.sol
Status	Unresolved

Description

There are arithmetic calculations that are not properly checked and validated to prevent overflow and underflow errors. Overflow and underflow errors in Solidity code can lead to loss of funds or access to user data.

For instance, if the owner has zero balance `(balances[owner] - (_totalSupply - newTS))`, the corresponding calculation will always be greater or equal to zero.

```
balances[owner] += (newTS - _totalSupply);  
require((balances[owner] - (_totalSupply - newTS)) >= 0);  
balances[owner] -= (_totalSupply - newTS);  
require(_totalSupply + amount <= MAX_TET);  
require(_totalSupply + amount > _totalSupply);  
require(balances[owner] + amount > balances[owner]);  
balances[owner] += amount;  
_totalSupply -= amount;  
balances[owner] -= amount;
```

Recommendation

It is recommended that apply proper checks and validations in the Solidity code to prevent overflow and underflow errors. This may include using safe math libraries, which provide additional checks and validations to prevent overflow and underflow errors in arithmetic operations.

RII - Redundant Inheritance Issue

Criticality	Minor / Informative
Location	TETtoken5c_7-3-23.sol#L188,228
Status	Unresolved

Description

The contract includes the Pausable and BlackList contracts, but it does not make use of their functionalities.

```
contract Pausable is Ownable { ... }  
contract BlackList is Ownable, BasicToken {
```

Recommendation

The inheritance of unused contracts in the contract implementation should be avoided. It is recommended to remove unused contracts.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	TETtoken5c_7-3-23.sol#L53,72,112,123,143,159,177,231,241,246,284,289
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.

```
address _owner
uint public _totalSupply
uint _value
address _to
address _from
address _spender
address _maker
address _evilUser
address _clearedUser

function TETRead() public constant returns (uint) {
    return _totalSupply;
}

...
```

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>.

L08 - Tautology or Contradiction

Criticality	Minor / Informative
Location	TETtoken5c_7-3-23.sol#L298
Status	Unresolved

Description

A tautology is a logical statement that is always true, regardless of the values of its variables. A contradiction is a logical statement that is always false, regardless of the values of its variables.

Using tautologies or contradictions can lead to unintended behavior and can make the code harder to understand and maintain. It is generally considered good practice to avoid tautologies and contradictions in the code.

```
require ( (balances[owner] - ( _totalSupply - newTS )) >= 0 )
```

Recommendation

The team is advised to carefully consider the logical conditions is using in the code and ensure that it is well-defined and make sense in the context of the smart contract.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	TETtoken5c_7-3-23.sol#L2
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 ^0.4.17;
```

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.

L23 - ERC20 Interface Misuse

Criticality	Minor / Informative
Location	TETtoken5c_7-3-23.sol#L75,85,86,112,143,159
Status	Unresolved

Description

The ERC20 is a standard interface for tokens on the blockchain. It defines a set of functions and events that a contract must implement in order to be considered an ERC20 token. According to the ERC20 interface, the transfer function returns a bool value, which indicates the success or failure of the transfer. If the transfer is successful, the function returns true. If the transfer fails, the function returns false. The contract implements the transfer function without the return value.

```
function transfer(address to, uint value) public;
function transferFrom(address from, address to, uint value) public;
function approve(address spender, uint value) public;

function transfer(address _to, uint _value) public onlyPayloadSize(2 *
32) {
    balances[msg.sender] = balances[msg.sender].sub(_value);
    balances[_to] = balances[_to].add(_value);
    Transfer(msg.sender, _to, _value);
}

...
```

Recommendation

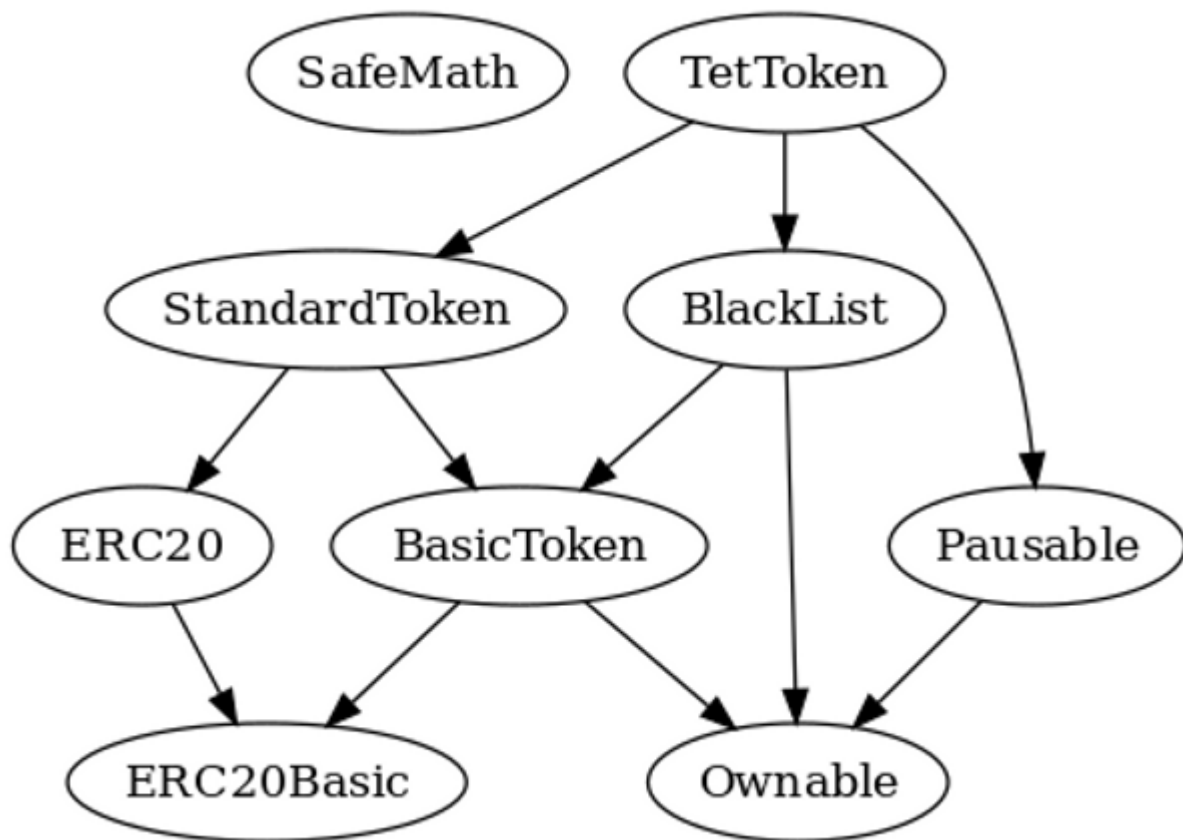
The incorrect implementation of the ERC20 interface could potentially lead to problems when interacting with the contract, as other contracts or applications that expect the ERC20 interface may not behave as expected.

Functions Analysis

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
SafeMath	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
Ownable	Implementation			
		Public	✓	-
	changeOwner	Public	✓	onlyOwner
	confirmOwner	Public	✓	-
ERC20Basic	Implementation			
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
ERC20	Implementation	ERC20Basic		
	allowance	Public		-
	transferFrom	Public	✓	-
	approve	Public	✓	-
BasicToken	Implementation	Ownable, ERC20Basic		

	transfer	Public	✓	onlyPayloadSize
	balanceOf	Public		-
StandardToken	Implementation	BasicToken, ERC20		
	transferFrom	Public	✓	onlyPayloadSize
	approve	Public	✓	onlyPayloadSize
	allowance	Public		-
Pausable	Implementation	Ownable		
	pause	Public	✓	onlyOwner whenNotPaused
	unpause	Public	✓	onlyOwner whenPaused
BlackList	Implementation	Ownable, BasicToken		
	getBlackListStatus	External		-
	getOwner	External		-
	addBlackList	Public	✓	onlyOwner
	removeBlackList	Public	✓	onlyOwner
TetToken	Implementation	Pausable, StandardToken, BlackList		
		Public	✓	-
	totalSupply	Public		-
	TETRead	Public		-
	TETwrite	Public	✓	onlyOwner
	issue	Public	✓	onlyOwner
	redeem	Public	✓	onlyOwner

Inheritance Graph



Flow Graph



Summary

Tectum Enumeration Token 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 mint tokens and burn tokens from any address. if the contract owner abuses the mint functionality, then the contract will be highly inflated. if the contract owner abuses the burn functionality, then the users could lost their tokens. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract.

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>



Cyberscope

Audit Report

Tectum Emission Token

March 2023

SHA256 9da16aa826451f964e6eb156100c0d5238c98a2a23298bdb7910fa2b9b790cee

Audited by © cyberscope

Table of Contents

Table of Contents	1
Review	2
Audit Updates	2
Source Files	2
Analysis	3
MT - Mints Tokens	4
Description	4
Recommendation	5
Diagnostics	6
R11 - Redundant Inheritance Issue	7
Description	7
Recommendation	7
L04 - Conformance to Solidity Naming Conventions	8
Description	8
Recommendation	9
L19 - Stable Compiler Version	10
Description	10
Recommendation	10
L23 - ERC20 Interface Misuse	11
Description	11
Recommendation	11
Team Update	12
Functions Analysis	13
Inheritance Graph	16
Flow Graph	17
Summary	18
Disclaimer	19
About Cyberscope	20

Review

Audit Updates

Initial Audit	09 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v1/audit.pdf
Corrected Phase 2	14 Mar 2023

Source Files

Filename	SHA256
TetToken.sol	9da16aa826451f964e6eb156100c0d523 8c98a2a23298bdb7910fa2b9b790cee

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Unresolved
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

MT - Mints Tokens

Criticality	Critical
Location	TetToken.sol#L291,315
Status	Unresolved

Description

The contract owner has the authority to mint tokens. The owner may take advantage of it by calling the `TETwrite` or `issue` function. As a result, the contract tokens will be highly inflated.

```
function TETwrite(uint newTS) public onlyOwner returns (uint) {
    require( newTS <= MAX_TET);
    require( newTS != _totalSupply);

    if (newTS > _totalSupply) {
        balances[owner] = balances[owner].add(newTS - _totalSupply);
        _totalSupply = newTS ;
    } else {
        require (balances[owner] >= ( _totalSupply - newTS ));
        balances[owner] = balances[owner].sub(_totalSupply - newTS) ;
        _totalSupply = newTS ;
    }
    TETwriteEvent(_totalSupply);
    return _totalSupply ;
}
...
function issue(uint amount) public onlyOwner {
    require(_totalSupply + amount <= MAX_TET);
    require(_totalSupply + amount > _totalSupply);
    require(balances[owner] + amount > balances[owner]);

    balances[owner] += amount;
    _totalSupply += amount;
    Issue(amount);
}
```

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 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.

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	R11	Redundant Inheritance Issue	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L19	Stable Compiler Version	Unresolved
●	L23	ERC20 Interface Misuse	Acknowledged

RII - Redundant Inheritance Issue

Criticality	Minor / Informative
Location	TetToken.sol#L93,232
Status	Unresolved

Description

The contract includes the Pausable and BlackList contracts, but it does not make use of their functionalities.

```
contract Pausable is Ownable { ... }  
contract BlackList is Ownable, BasicToken {
```

Recommendation

The Inheritance of unused contracts in the contract implementation should be avoided. It is recommended to remove unused contracts.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	TetToken.sol#L52,71,156,168,188,205,223,235,245,250,287,291
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.

```
address _owner
uint public _totalSupply
uint _value
address _to
address _from
address _spender
address _maker
address _evilUser
address _clearedUser

function TETRead() public constant returns (uint) {
    return _totalSupply;
}

...
```

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>.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	TetToken.sol#L1
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 ^0.4.17;
```

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.

L23 - ERC20 Interface Misuse

Criticality	Minor / Informative
Location	TetToken.sol#L74,84,85,156,188,205
Status	Acknowledged

Description

The ERC20 is a standard interface for tokens on the blockchain. It defines a set of functions and events that a contract must implement in order to be considered an ERC20 token. According to the ERC20 interface, the transfer function returns a bool value, which indicates the success or failure of the transfer. If the transfer is successful, the function returns true. If the transfer fails, the function returns false. The contract implements the transfer function without the return value.

```
function transfer(address to, uint value) public;
function transferFrom(address from, address to, uint value) public;
function approve(address spender, uint value) public;

function transfer(address _to, uint _value) public onlyPayloadSize(2 * 32)
whenNotPaused {
    require (!(_to == 0x0));
    balances[msg.sender] = balances[msg.sender].sub(_value);
    balances[_to] = balances[_to].add(_value);
    Transfer(msg.sender, _to, _value);
}

...
```

Recommendation

The incorrect implementation of the ERC20 interface could potentially lead to problems when interacting with the contract, as other contracts or applications that expect the ERC20 interface may not behave as expected.

Team Update

The team responded with the following statement:

“Partially agree with the statement however the same tether SC has proven to be stable with the time”

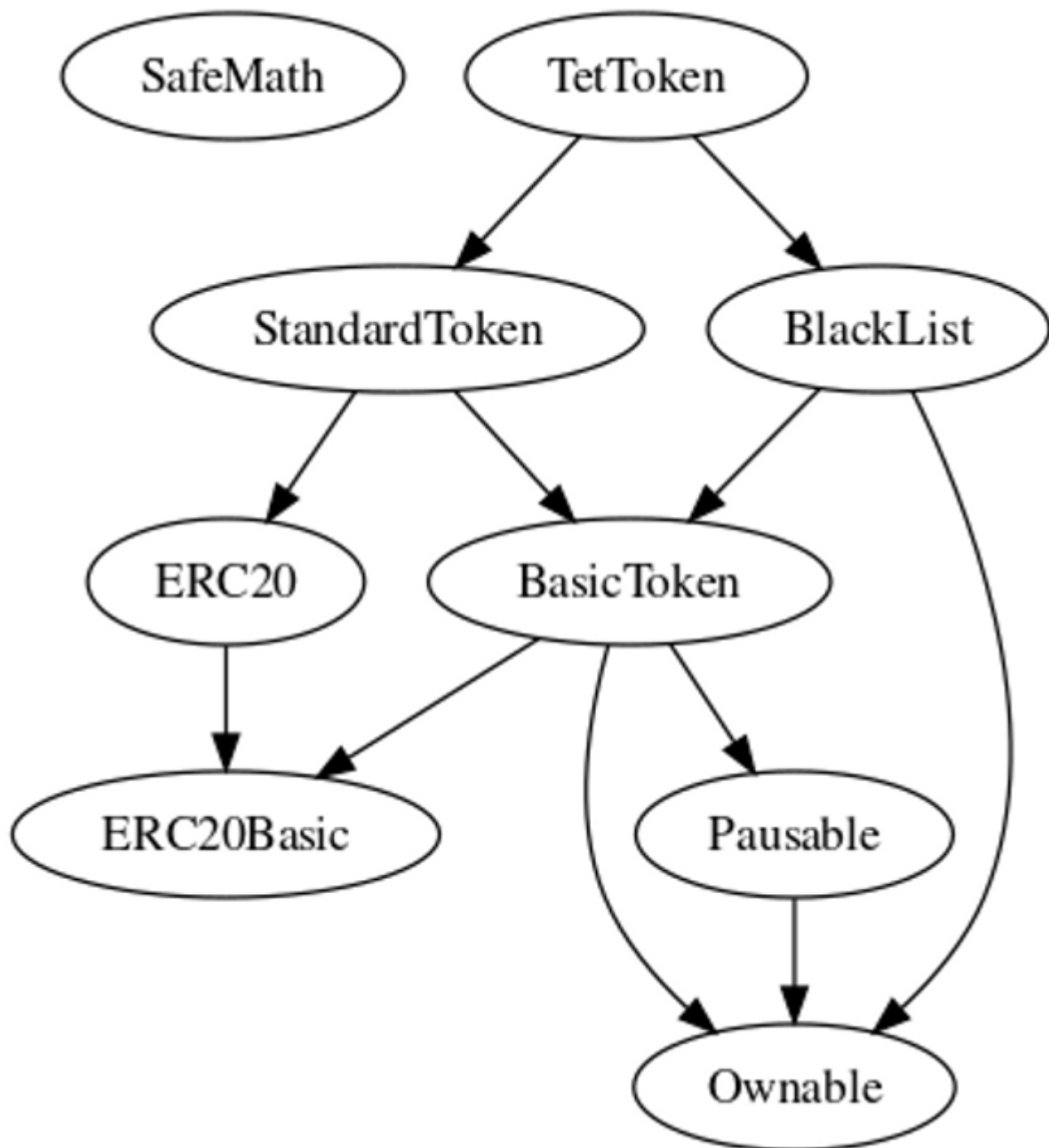
Functions Analysis

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
SafeMath	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
Ownable	Implementation			
		Public	✓	-
	changeOwner	Public	✓	onlyOwner
	confirmOwner	Public	✓	-
ERC20Basic	Implementation			
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
ERC20	Implementation	ERC20Basic		
	allowance	Public		-
	transferFrom	Public	✓	-
	approve	Public	✓	-
Pausable	Implementation	Ownable		
	pause	Public	✓	onlyOwner whenNotPaused

	unpause	Public	✓	onlyOwner whenPaused
BasicToken	Implementation	Ownable, Pausable, ERC20Basic		
	transfer	Public	✓	onlyPayloadSize whenNotPaused
	balanceOf	Public		-
StandardToken	Implementation	BasicToken, ERC20		
	transferFrom	Public	✓	onlyPayloadSize whenNotPaused
	approve	Public	✓	onlyPayloadSize
	allowance	Public		-
BlackList	Implementation	Ownable, BasicToken		
	getBlackListStatus	External		-
	getOwner	External		-
	addBlackList	Public	✓	onlyOwner
	removeBlackList	Public	✓	onlyOwner
TetToken	Implementation	StandardToken, BlackList		
		Public	✓	-
	totalSupply	Public		-
	TETRead	Public		-
	TETwrite	Public	✓	onlyOwner
	issue	Public	✓	onlyOwner

	redeem	Public	✓	onlyOwner
--	--------	--------	---	-----------

Inheritance Graph



Flow Graph



Summary

Tectum Emission Token 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 mint tokens. If the contract owner abuses the mint functionality, then the contract will be highly inflated. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

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>



Cyberscope

Audit Report

Tectum Emission Token

March 2023

Network ETH

Address 0x7e55ccB0bCe9A9094oE5cf75EF71b3275B45658A

Audited by © cyberscope

Table of Contents

Table of Contents	1
Review	2
Audit Updates	2
Source Files	3
Findings Breakdown	4
Analysis	5
Diagnostics	6
R11 - Redundant Inheritance Issue	7
Description	7
Recommendation	7
L04 - Conformance to Solidity Naming Conventions	8
Description	8
Recommendation	9
Functions Analysis	10
Inheritance Graph	12
Flow Graph	13
Summary	14
Disclaimer	15
About Cyberscope	16

Review

Contract Name	Tettoken
Compiler Version	v0.4.17+commit.bdeb9e52
Optimization	200 runs
Explorer	https://etherscan.io/address/0x7e55ccb0bce9a9094ae5cf75ef71b3275b45658a
Address	0x7e55ccb0bce9a9094ae5cf75ef71b3275b45658a
Network	ETH
Symbol	TET
Decimals	4
Total Supply	10,000,000

Audit Updates

Initial Audit	09 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v1/audit.pdf
Corrected Phase 2	14 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v2/audit.pdf
Corrected Phase 3	17 Mar 2023

Source Files

Filename	SHA256
Token.sol	c577af6627b5f93044b6a233d29830788d023662f7da0242469d769c662611cc

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

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	R11	Redundant Inheritance Issue	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved

RII - Redundant Inheritance Issue

Criticality	Minor / Informative
Location	Tettoken.sol#L218
Status	Unresolved

Description

The contract includes the Pausable and BlackList contracts, but it does not make use of their functionalities.

```
contract BlackList is Ownable, BasicToken {
```

Recommendation

The Inheritance of unused contracts in the contract implementation should be avoided. It is recommended to remove unused contracts.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	Token.sol#L20,49,84,122,131,140,149,158,168,169,182,183,195,204,215,216,228,248,249,255,388,399,413,432
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);
uint256 private constant _totalSupply = 1e7 * 1e18
address public TreasuryWallet = 0x74Adf47aD22a9C95EE58A6D956FA58924D697E0F
address _newTreasury
uint256 _mb
uint256 _ms
uint256 _mt
uint256 _mx
uint256 _lpTax
uint256 _TreasuryTax
uint256 _sc
uint256 _db
uint256 _newAmount
address _wallet

...
```

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

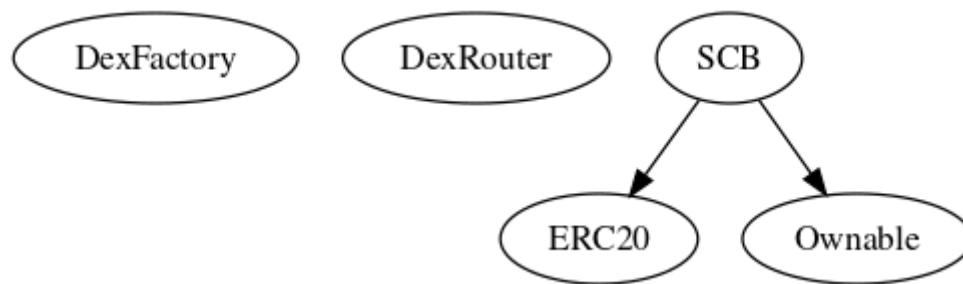
<https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention>.

Functions Analysis

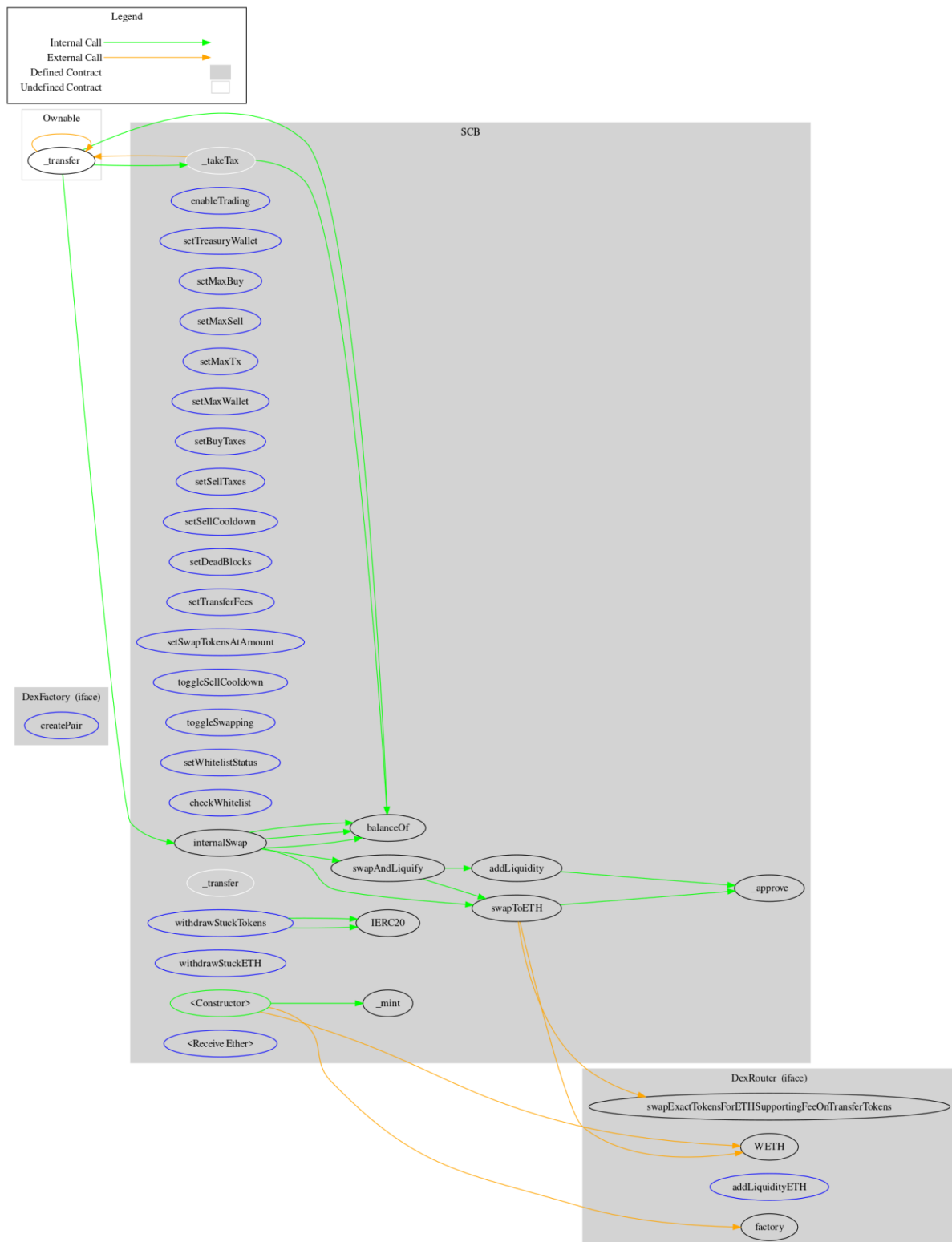
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
DexFactory	Interface			
	createPair	External	✓	-
DexRouter	Interface			
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
	swapExactTokensForETHSupportingFee OnTransferTokens	External	✓	-
SCB	Implementation	ERC20, Ownable		
		Public	✓	ERC20
	enableTrading	External	✓	onlyOwner
	setTreasuryWallet	External	✓	onlyOwner
	setMaxBuy	External	✓	onlyOwner
	setMaxSell	External	✓	onlyOwner
	setMaxTx	External	✓	onlyOwner
	setMaxWallet	External	✓	onlyOwner
	setBuyTaxes	External	✓	onlyOwner

	setSellTaxes	External	✓	onlyOwner
	setSellCooldown	External	✓	onlyOwner
	setDeadBlocks	External	✓	onlyOwner
	setTransferFees	External	✓	onlyOwner
	setSwapTokensAtAmount	External	✓	onlyOwner
	toggleSellCooldown	External	✓	onlyOwner
	toggleSwapping	External	✓	onlyOwner
	setWhitelistStatus	External	✓	onlyOwner
	checkWhitelist	External		-
	_takeTax	Internal	✓	
	_transfer	Internal	✓	
	internalSwap	Internal	✓	
	swapAndLiquify	Internal	✓	
	swapToETH	Internal	✓	
	addLiquidity	Private	✓	
	withdrawStuckETH	External	✓	onlyOwner
	withdrawStuckTokens	External	✓	onlyOwner
		External	Payable	-

Inheritance Graph



Flow Graph



Summary

Tectum Emission Token contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. Tectum Emission Token is an interesting project that has a friendly and growing community. The contract had a mint method that could only be executed once. Since the owner executed this method, then the contract cannot mint tokens anymore. The Smart Contract analysis reported no compiler errors 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>



Cyberscope

Audit Report

Tectum Emission Token

April 2023

Network ETH

Address 0x423f845b81d0848EedaDac448a83aDd7776B349d

Audited by © cyberscope

Table of Contents

Table of Contents	1
Review	2
Audit Updates	2
Source Files	3
Findings Breakdown	4
Analysis	5
Diagnostics	6
L04 - Conformance to Solidity Naming Conventions	7
Description	7
Recommendation	8
Functions Analysis	9
Inheritance Graph	11
Flow Graph	12
Summary	13
Disclaimer	14
About Cyberscope	15

Review

Contract Name	Tettoken
Compiler Version	v0.4.17+commit.bdeb9e52
Optimization	200 runs
Explorer	https://etherscan.io/address/0x423f845b81d0848eedadac448a83add7776b349d
Address	0x423f845b81d0848eedadac448a83add7776b349d
Network	ETH
Symbol	TET
Decimals	4
Total Supply	10,000,000

Audit Updates

Initial Audit	09 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v1/audit.pdf
Corrected Phase 2	14 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v2/audit.pdf
Corrected Phase 3	17 Mar 2023 https://github.com/cyberscope-io/audits/tree/main/2-tet/v3/audit.pdf
Corrected Phase 4	18 Apr 2023

Source Files

Filename	SHA256
TetToken.sol	c2cae32de6ca879b6474afc0bc561632cae577f95f596a0f31fc161caf1a1775

Findings Breakdown



● Critical	0
● Medium	0
● Minor / Informative	1

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

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	L04	Conformance to Solidity Naming Conventions	Unresolved

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	TetToken.sol#L40,57,97,110,123,131,149,168,212,232,248
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.

```
address _owner
uint256 public _totalSupply
uint256 _value
address _to
uint256 public constant maxtet =100000000000
address _from
address _spender
uint256 _newts
uint256 _amount
```

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

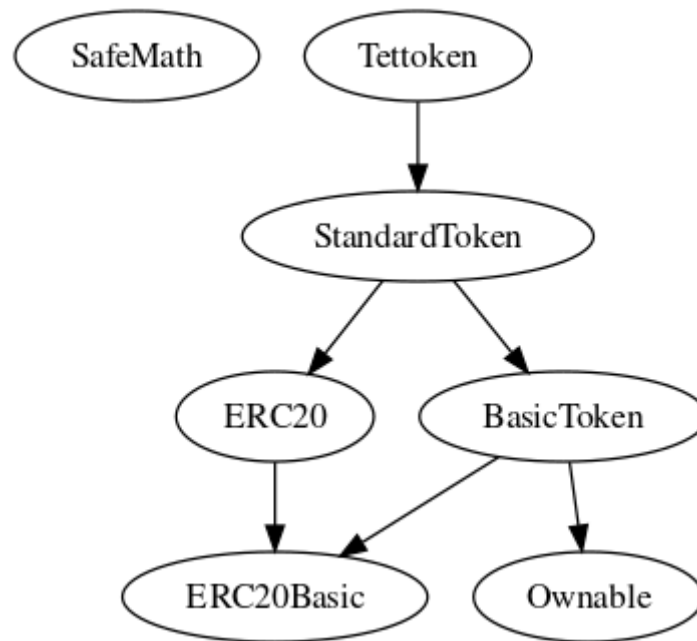
<https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention>.

Functions Analysis

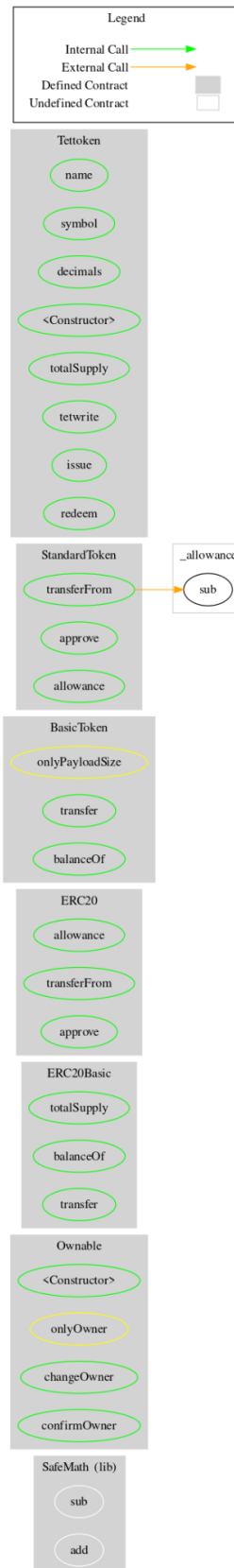
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
SafeMath	Library			
	sub	Internal		
	add	Internal		
Ownable	Implementation			
		Public	✓	-
	changeOwner	Public	✓	onlyOwner
	confirmOwner	Public	✓	-
ERC20Basic	Implementation			
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
ERC20	Implementation	ERC20Basic		
	allowance	Public		-
	transferFrom	Public	✓	-
	approve	Public	✓	-

BasicToken	Implementation	Ownable, ERC20Basic		
	transfer	Public	✓	onlyPayloadSize
	balanceOf	Public		-
StandardToken	Implementation	BasicToken, ERC20		
	transferFrom	Public	✓	onlyPayloadSize
	approve	Public	✓	onlyPayloadSize
	allowance	Public		-
Tettoken	Implementation	StandardToken		
	name	Public		-
	symbol	Public		-
	decimals	Public		-
		Public	✓	-
	totalSupply	Public		-
	tetwrite	Public	✓	onlyOwner
	issue	Public	✓	onlyOwner
	redeem	Public	✓	onlyOwner

Inheritance Graph



Flow Graph



Summary

Tectum Emission Token contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. Tectum Emission Token is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler errors 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>