Cyberscope

Audit Report

Tails

August 2024

Repository https://github.com/zbagdzevicius/tokentails/tree/main

Commit 8ea0b86e2c305c25ba4b2baa91b5e710d239e5de

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	Unresolved
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	IDI	Immutable Declaration Improvement	Unresolved
●	MVO	MaxSupply Variable Optimization	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	TokenTails
Repository	https://github.com/zbagdzevicius/tokentails/tree/main
Commit	8ea0b86e2c305c25ba4b2baa91b5e710d239e5de
Testing Deploy	https://testnet.bscscan.com/address/0x6e9fc2b9fd134574f099c01deb2bdc920b2b3677
Symbol	TAILS
Decimals	18
Badge Eligibility	Must Fix Criticals

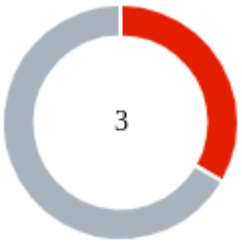
Audit Updates

Initial Audit	06 Aug 2024

Source Files

Filename	SHA256
contracts/Tails.sol	5be8c1ee477076cdc14e81284c6220ce0db81e584b3238d15b55b2e7ae3bd95e

Findings Breakdown



Severity	Unresolved	Acknowledged	Resolved	Other
Critical	1	0	0	0
Medium	0	0	0	0
Minor / Informative	2	0	0	0

MT - Mints Tokens

Criticality	Critical
Location	contracts/Tails.sol#L25
Status	Unresolved

Description

The `MINTER_ROLE` has the authority to mint tokens. The `MINTER_ROLE` may take advantage of it by calling the `mint` function, until the `_maxSupply` amount is reached. As a result the contract tokens will be highly inflated.

```
    function mint(address to, uint256 amount) public
onlyRole(MINTER_ROLE) {
        uint256 amountWithDecimals = amount * 10 ** decimals();
// Adjust amount for decimals
        require(totalSupply() + amountWithDecimals <=
_maxSupply, "ERC20: minting would exceed max supply");
        _mint(to, amountWithDecimals);
    }
```

Recommendation

The team should carefully manage the private keys of the `MINTER_ROLE` account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

IDI - Immutable Declaration Improvement

Criticality	Minor / Informative
Location	contracts/Tails.sol#L18
Status	Unresolved

Description

The contract declares state variables that their value is initialized once in the constructor and are not modified afterwards. The `immutable` is a special declaration for this kind of state variables that saves gas when it is defined.

```
_maxSupply
```

Recommendation

By declaring a variable as immutable, the Solidity compiler is able to make certain optimizations. This can reduce the amount of storage and computation required by the contract, and make it more gas-efficient.

MVO - MaxSupply Variable Optimization

Criticality	Minor / Informative
Location	contracts/Tails.sol#L11
Status	Unresolved

Description

The contract is using the `maxSupply()` function solely to return the `_maxSupply` value. This introduces unnecessary complexity by adding an extra function call where the same result can be achieved more efficiently. Instead of implementing a separate function, the `_maxSupply` variable could be declared as `public`, allowing direct access to its value. This would streamline the contract, reducing gas costs and improving readability.

```
    uint256 private _maxSupply;

    constructor()
        ERC20("Token Tails", "TAILS")
    {
        _grantRole(DEFAULT_ADMIN_ROLE, _msgSender());
        _grantRole(MINTER_ROLE, _msgSender());
        _maxSupply = 500000000 * 10 ** decimals(); // Setting max
supply to 500,000,000 with decimals adjustment
    }

    function maxSupply() public view returns (uint256) {
        return _maxSupply;
    }
```

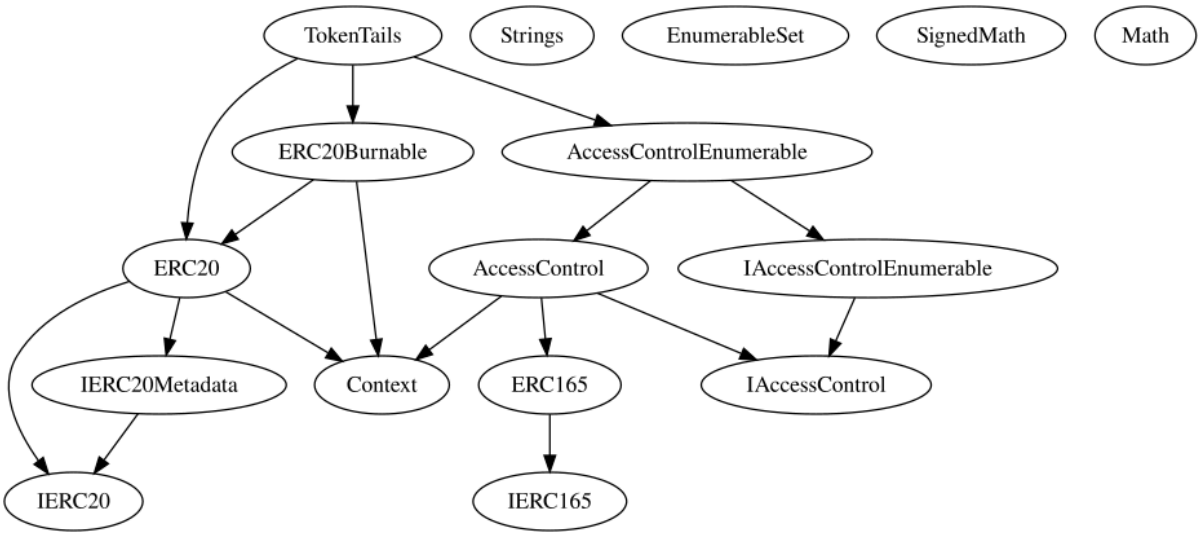
Recommendation

It is recommended to declare the `_maxSupply` variable as `public` to eliminate the need for the `maxSupply` function. This change will simplify the contract, making it more efficient by reducing gas consumption and enhancing overall clarity.

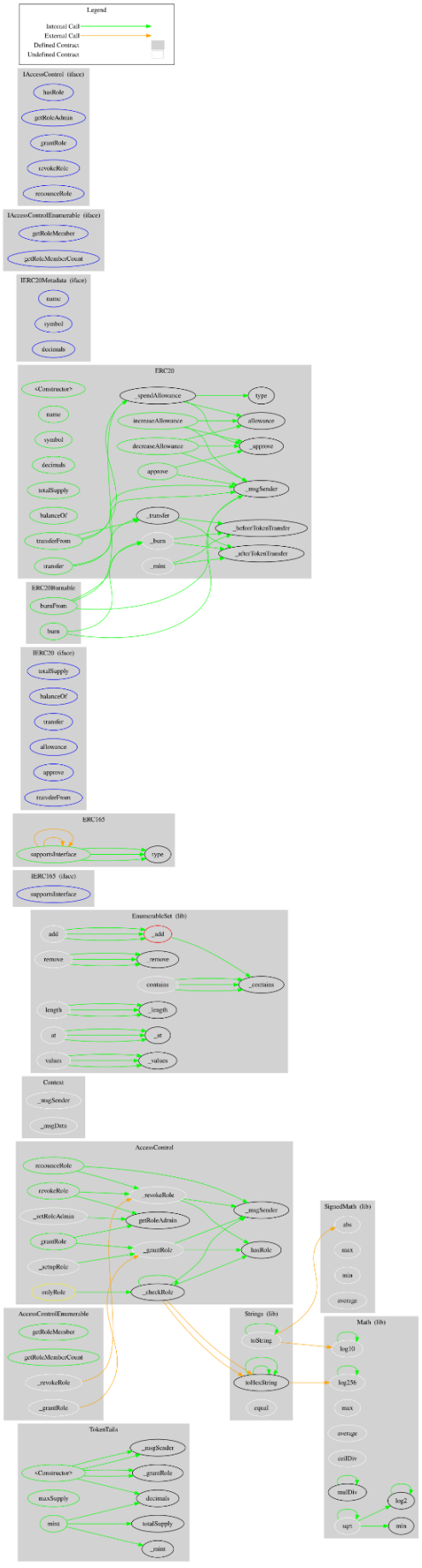
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
TokenTails	Implementation	ERC20, ERC20Burnable, AccessControlEnumerable		
		Public	✓	ERC20
	maxSupply	Public		-
	mint	Public	✓	onlyRole

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IAccessControl (iface)
hasRole
getRoleAdmin
grantRole
revokeRole
renounceRole
IAccessControlEnumerable (iface)
getRoleMember
getRoleMemberCount
IERC20Metadata (iface)
name
symbol
decimals
ERC20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
transferFrom
transfer
_spendAllowance
increaseAllowance
decreaseAllowance
approve
_transfer
_burn
_mint
type
allowance
_approve
_msgSender
_beforeTokenTransfer
_afterTokenTransfer
ERC20Burnable
burnFrom
burn
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
IERC165
supportsInterface
type
IERC165 (iface)
supportsInterface
EnumerableSet (lib)
add
remove
contains
length
at
values
_add
_remove
_contains
_length
_at
_values
Context
_msgSender
_msgData
AccessControl
renounceRole
revokeRole
_setRoleAdmin
grantRole
_setupRole
onlyRole
_revokeRole
getRoleAdmin
_grantRole
_checkRole
_msgSender
hasRole
SignedMath (lib)
abs
max
min
average
AccessControlEnumerable
getRoleMember
getRoleMemberCount
_revokeRole
_grantRole
Strings (lib)
toString
toHexString
equal
Math (lib)
log10
log256
max
average
ceilDiv
mulDiv
sqrt
log2
mix
TokenTails
<Constructor>
maxSupply
mint
_msgSender
_grantRole
decimals
totalSupply
_mint

Summary

Tails 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

cyberscope.io