Cyberscope

Audit Report

ParadiseChain

April 2024

SHA256 7df5ee5c670f4e6bb9743c2886bb0af1b1de9bd07c62f9b1a3fc220c219e5e05

Audited by © cyberscope

Table of Contents

Review

Contract Name	Token
Testing Deploy	https://testnet.bscscan.com/address/0x2322fa94cbdc2f8780942083cfe1d07cdc2b4d5a
Symbol	TD
Decimals	18

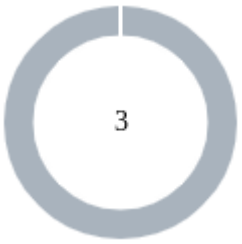
Audit Updates

Initial Audit	14 May 2024

Source Files

Filename	SHA256
contracts/Token.sol	7df5ee5c670f4e6bb9743c2886bb0af1b1de9bd07c62f9b1a3fc220c219e5e05

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

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	CLU	Console Log Usage	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L19	Stable Compiler Version	Unresolved

CLU - Console Log Usage

Criticality	Minor / Informative
Location	contracts/ParadiseChainToken.sol#L7
Status	Unresolved

Description

The contract is currently utilizing the `hardhat/console.log` import within its codebase, a practice typically reserved for development and testing phases to debug and verify contract logic. While `console.log` is an invaluable tool for developers to trace and debug contract execution in a test environment, its presence in a contract intended for deployment on the mainnet (production environment) raises concerns. The primary issue is that these logging statements are part of the Hardhat environment and do not function in a live network. Their inclusion in a production-ready contract suggests that the contract may not have been adequately cleaned up post-testing, potentially indicating oversight in the development process.

```
import "hardhat/console.sol";
```

Recommendation

It is recommended to remove the `hardhat/console.log` import from the contract before deployment to the mainnet or any production environment. This cleanup step is crucial since it reduces the contract's bytecode size, thereby lowering deployment and execution costs, and it eliminates unnecessary code that serves no purpose in a live environment.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	contracts/ParadiseChainToken.sol#L17
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 public TGETimestamp
```

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	contracts/ParadiseChainToken.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 ^0.8.20;
```

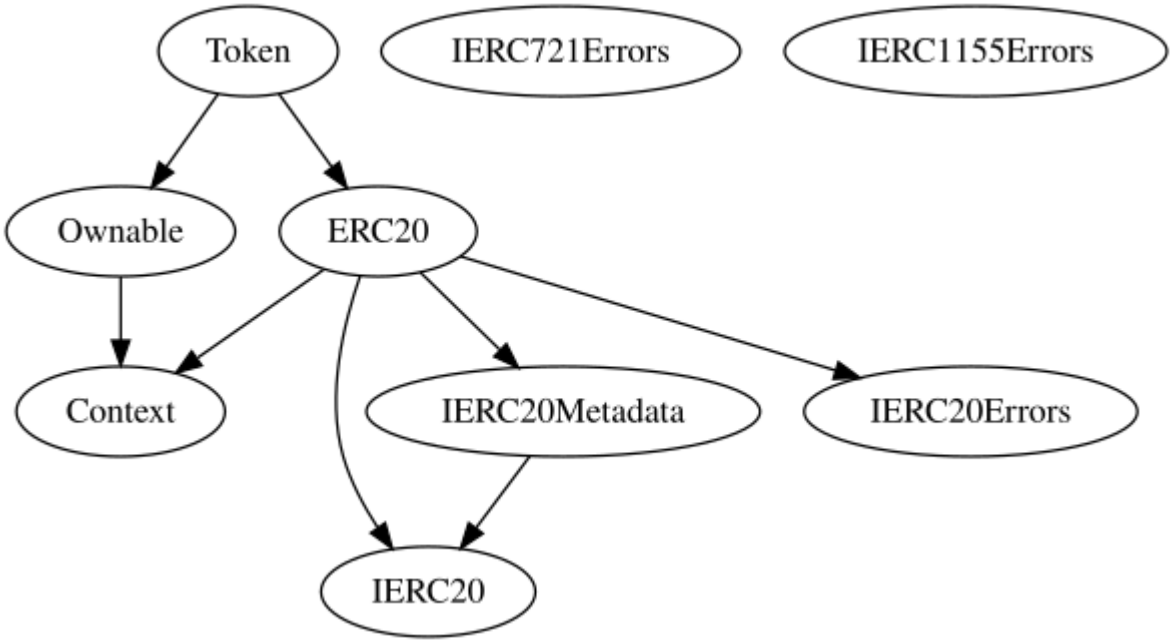
Recommendation

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.

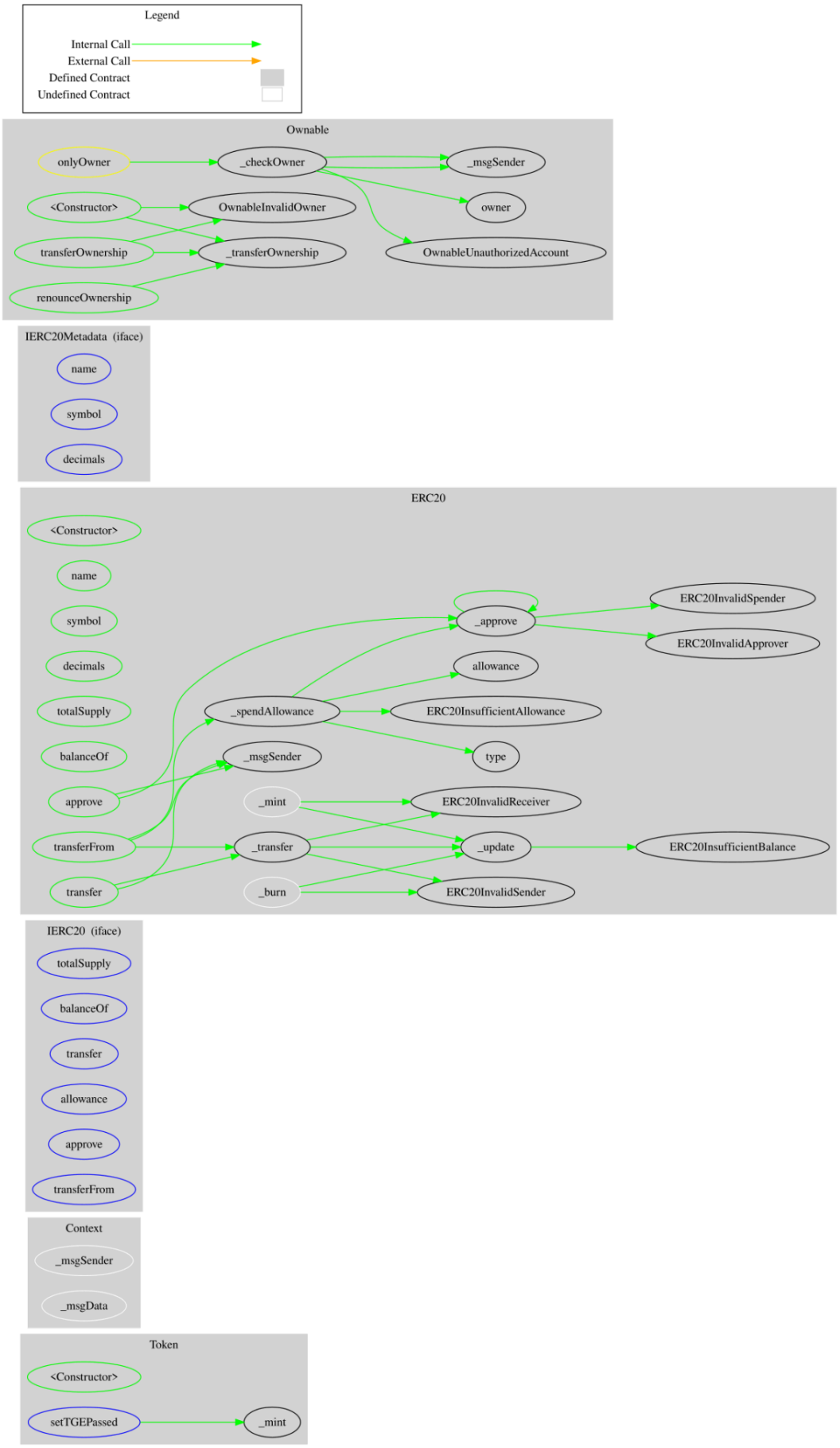
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Token	Implementation	ERC20, Ownable		
		Public	✓	ERC20 Ownable
	setTGEPassed	External	✓	onlyOwner

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Ownable
onlyOwner
_checkOwner
_msgSender
<Constructor>
OwnableInvalidOwner
owner
transferOwnership
_transferOwnership
OwnableUnauthorizedAccount
renounceOwnership
IERC20Metadata (iface)
name
symbol
decimals
ERC20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
approve
transferFrom
transfer
_approve
ERC20InvalidSpender
ERC20InvalidApprover
allowance
_spendAllowance
ERC20InsufficientAllowance
type
_msgSender
_mint
ERC20InvalidReceiver
_transfer
_update
ERC20InsufficientBalance
_burn
ERC20InvalidSender
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Context
_msgSender
_msgData
Token
<Constructor>
setTGEPassed
_mint

Summary

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