Cyberscope

Audit Report

CYBER

August 2024

Network	BSC
Address	0xaeD5a3253695f5c4A4674328EFb3936a9938ac0F
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
●	EFL	Exceeds Fees Limit	Unresolved
●	MEE	Missing Events Emission	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L08	Tautology or Contradiction	Unresolved
●	L16	Validate Variable Setters	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	Cyberware
Compiler Version	v0.8.26+commit.8a97fa7a
Optimization	300 runs
Explorer	https://bscscan.com/address/0xaed5a3253695f5c4a4674328efb3936a9938ac0f
Address	0xaed5a3253695f5c4a4674328efb3936a9938ac0f
Network	BSC
Decimals	18

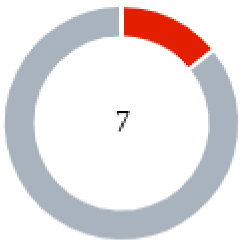
Audit Updates

Initial Audit	27 Aug 2024

Source Files

Filename	SHA256
contracts/CyberwareToken.sol	ee1e2b2458f21911424706cda9e75cae0433d6380c57dc78fd2467898dbef762

Findings Breakdown



●	Critical	1
●	Medium	0
●	Minor / Informative	6

Severity		Unresolved	Acknowledged	Resolved	Other
●	Critical	1	0	0	0
●	Medium	0	0	0	0
●	Minor / Informative	6	0	0	0

MT - Mints Tokens

Criticality	Critical
Location	contracts/CyberwareToken.sol#L43
Status	Unresolved

Description

The address possessing the `MINTER_ROLE` has the authority to mint tokens. The minter may take advantage of it by calling the `mint` function. As a result, the contract tokens will be highly inflated.

```
function mint(address to, uint256 amount) public
onlyRole(MINTER_ROLE) {
        _mint(to, amount);
    }
```

Notably, the address possessing the `DEFAULT_ADMIN_ROLE` has the ability to grant or revoke the `MINTER_ROLE` to or from any address at any time.

Recommendation

The team should carefully manage the private keys of the minting and administrator accounts. 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 minter and administrator roles.

EFL - Exceeds Fees Limit

Criticality	Minor / Informative
Location	contracts/CyberwareToken.sol#L47,57
Status	Unresolved

Description

The contract includes a fee mechanism applied during the burning of tokens. The administrator can set the `BURN_FEE` variable to any percentage up to 100%. This could be misleading for users, as a substantial portion of the tokens are not burned but instead transferred to the team's wallets.

```
function burn(uint256 amount) public
override(ERC20BurnableUpgradeable) {
        if (msg.sender != TREASURY_ADDRESS && BURN_FEE > 0) {
            uint256 treasuryBurnFee = amount * BURN_FEE /
10000;
            amount -= treasuryBurnFee;
            _transfer(msg.sender, TREASURY_ADDRESS,
treasuryBurnFee);
        }

        _burn(msg.sender, amount);
}

function setBurnFee(uint256 amount) public
onlyRole(DEFAULT_ADMIN_ROLE) {
        require(amount <= 10000 && amount >=0,
"Cyberware::setBurnFee: AMOUNT_OUT_OF_RANGE");
        BURN_FEE = amount;
}
```

Recommendation

The contract could embody a check for the maximum acceptable value. The team should carefully manage the private keys of the administrator's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing these 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 administrator role.

MEE - Missing Events Emission

Criticality	Minor / Informative
Location	contracts/CyberwareToken.sol#L57,62
Status	Unresolved

Description

The contract performs actions and state mutations from external methods that do not result in the emission of events. Emitting events for significant actions is important as it allows external parties, such as wallets or dApps, to track and monitor the activity on the contract. Without these events, it may be difficult for external parties to accurately determine the current state of the contract.

```
function setBurnFee(uint256 amount) public
onlyRole(DEFAULT_ADMIN_ROLE) {
        require(amount <= 10000 && amount >=0,
"Cyberware::setBurnFee: AMOUNT_OUT_OF_RANGE");
        BURN_FEE = amount;
}

function setTreasuryAddress(address treasury) public
onlyRole(DEFAULT_ADMIN_ROLE) {
        require(treasury != address(0) && TREASURY_ADDRESS !=
treasury, "Cyberware::setTreasuryAddress: ADDRESS_INVALID");
        TREASURY_ADDRESS = treasury;
}
```

Recommendation

It is recommended to include events in the code that are triggered each time a significant action is taking place within the contract. These events should include relevant details such as the user's address and the nature of the action taken. By doing so, the contract will be more transparent and easily auditable by external parties. It will also help prevent potential issues or disputes that may arise in the future.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	contracts/CyberwareToken.sol#L17,18
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 BURN_FEE = 0
address public TREASURY_ADDRESS
```

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.

L08 - Tautology or Contradiction

Criticality	Minor / Informative
Location	contracts/CyberwareToken.sol#L58
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(amount <= 10000 && amount >=0, "Cyberware::setBurnFee:
AMOUNT_OUT_OF_RANGE")
```

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.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	contracts/CyberwareToken.sol#L40
Status	Unresolved

Description

The contract performs operations on variables that have been configured on user-supplied input. These variables are missing of proper check for the case where a value is zero. This can lead to problems when the contract is executed, as certain actions may not be properly handled when the value is zero.

```
TREASURY_ADDRESS = treasury
```

Recommendation

By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	contracts/CyberwareToken.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 ^0.8.26;
```

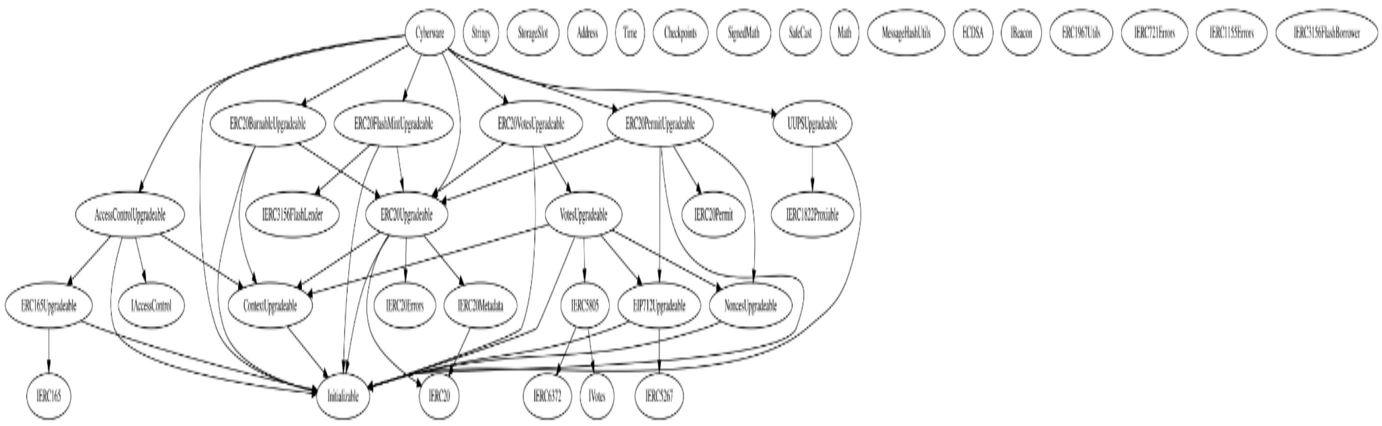
Recommendation

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.

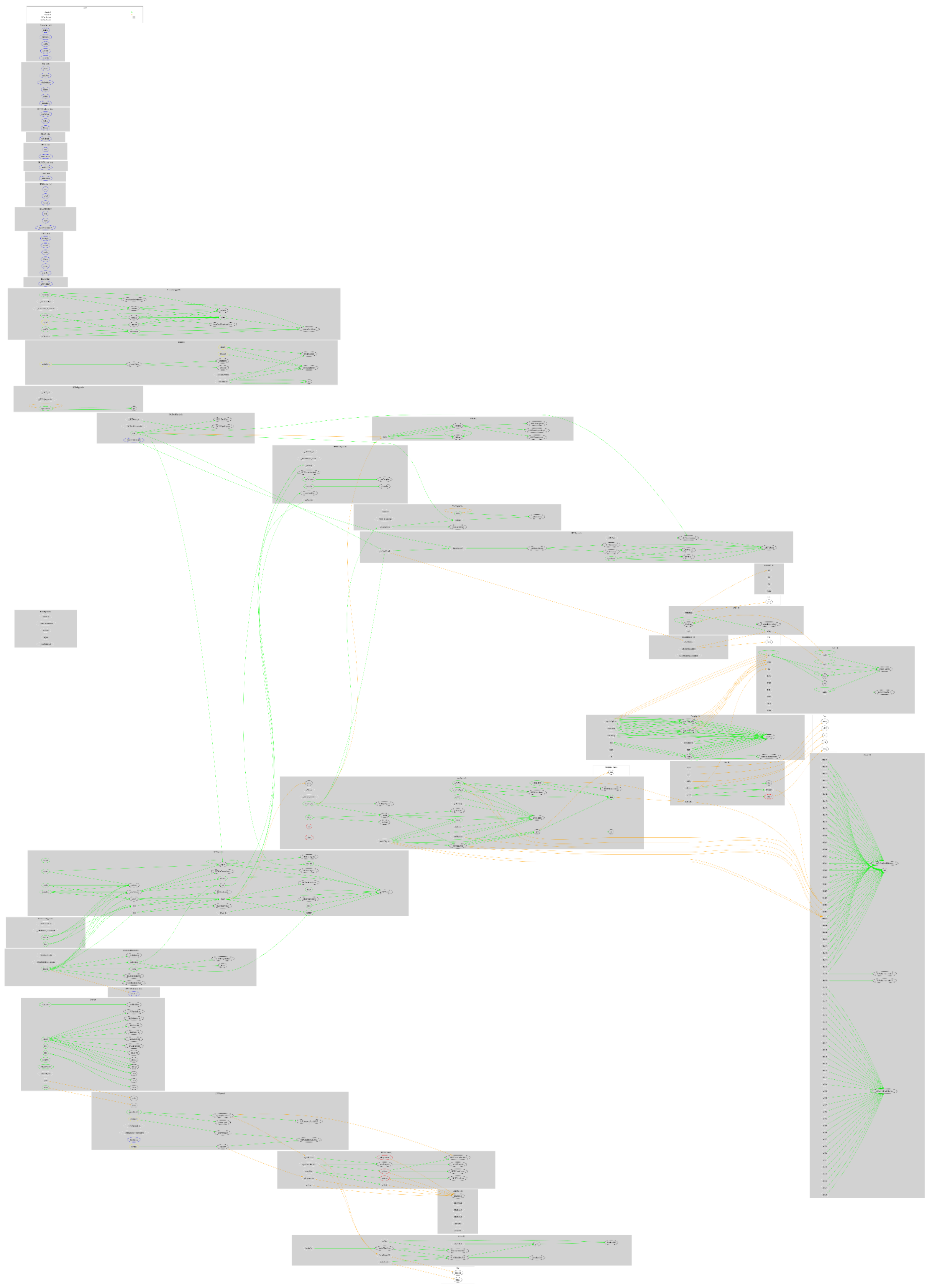
Functions Analysis

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Cyberware	Implementation	Initializable, ERC20Upgradeable, ERC20BurnableUpgradeable, AccessControlUpgradeable, ERC20PermitUpgradeable, ERC20VotesUpgradeable, ERC20FlashMintUpgradeable, UUPSUpgradeable		
		Public	✓	-
	initialize	Public	✓	initializer
	mint	Public	✓	onlyRole
	burn	Public	✓	-
	setBurnFee	Public	✓	onlyRole
	setTreasuryAddress	Public	✓	onlyRole
	_authorizeUpgrade	Internal	✓	onlyRole
	_update	Internal	✓	
	nonces	Public		-

Inheritance Graph



Flow Graph



Summary

CYBER 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 administrator and minter, like minting tokens. If these roles abuse the mint functionality, then the contract will be highly inflated. A multi-wallet signing pattern will provide security against potential hacks. Renouncing administrating and minting rights will eliminate the threat.

Initial Audit, 27 Aug 2024

At the time of the audit report, the contract with address 0xaeD5a3253695f5c4A4674328EFb3936a9938ac0F is pointed by the following proxy address: 0xaDc214418fb178F13A60588aaCc108f119fF70E7.

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