Cyberscope

Audit Report

CIPHER EPAY

October 2024

Network	MATIC
Address	0xBb4e25C1F0FbCB60A0d12456832412 65F1F64F61
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
●	CR	Code Repetition	Unresolved
●	IDI	Immutable Declaration Improvement	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L16	Validate Variable Setters	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	CipherEpay
Compiler Version	v0.8.17+commit.8df45f5f
Optimization	200 runs
Explorer	https://polygonscan.com/address/0xbb4e25c1f0fbcb60a0d1245683241265f1f64f61
Address	0xbb4e25c1f0fbcb60a0d1245683241265f1f64f61
Network	MATIC
Symbol	CPAY
Decimals	18
Total Supply	10,800,000,000
Badge Eligibility	Yes

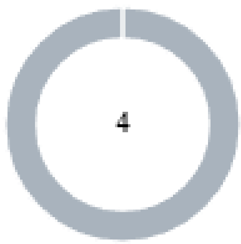
Audit Updates

Initial Audit	28 Oct 2024

Source Files

Filename	SHA256
ReflectiveERC20.sol	b97fdd2e79db14d55b875e3341ab2fa1f47c197dbb81484e6400e4856a168dba
CipherEpay.sol	0cdd95326fdc5d54148411b29c049c5e6c656cdb0809423b50044741d3ed8b3a
lib/LibCommon.sol	36b26da79703d916d5374759d61cd3b34897aa1f081b608bf9ce90d7fc43e866

Findings Breakdown



Critical	0
Medium	0
Minor / Informative	4

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	0	0	0
Minor / Informative	4	0	0	0

CR - Code Repetition

Criticality	Minor / Informative
Location	CipherEpay.sol#L317,346
Status	Unresolved

Description

The contract contains repetitive code segments. There are potential issues that can arise when using code segments in Solidity. Some of them can lead to issues like gas efficiency, complexity, readability, security, and maintainability of the source code. It is generally a good idea to try to minimize code repetition where possible. Specifically, the `transfer` and `transferFrom` functions share similar code segments.

```
function transfer(
    address to,
    uint256 amount
  ) public virtual override returns (bool) {
    uint256 taxAmount = _taxAmount(msg.sender, amount);
    uint256 deflationAmount = _deflationAmount(amount);
    uint256 amountToTransfer = amount - taxAmount -
deflationAmount;

    if (isMaxAmountOfTokensSet()) {
      if (balanceOf(to) + amountToTransfer >
maxTokenAmountPerAddress) {
        revert DestBalanceExceedsMaxAllowed(to);
      }
    }

    if (taxAmount != 0) {
      _transferNonReflectedTax(msg.sender, taxAddress,
taxAmount);
    }
    if (deflationAmount != 0) {
      _burn(msg.sender, deflationAmount);
    }
    return super.transfer(to, amountToTransfer);
  }

function transferFrom(
    address from,
    address to,
    uint256 amount
  ) public virtual override returns (bool) {
    uint256 taxAmount = _taxAmount(from, amount);
    uint256 deflationAmount = _deflationAmount(amount);
    uint256 amountToTransfer = amount - taxAmount -
deflationAmount;

    if (isMaxAmountOfTokensSet()) {
      if (balanceOf(to) + amountToTransfer >
maxTokenAmountPerAddress) {
        revert DestBalanceExceedsMaxAllowed(to);
      }
    }

    if (taxAmount != 0) {
      _transferNonReflectedTax(from, taxAddress, taxAmount);
    }
    if (deflationAmount != 0) {
      _burn(from, deflationAmount);
    }
```

```
    return super.transferFrom(from, to, amountToTransfer);
}
```

Recommendation

The team is advised to avoid repeating the same code in multiple places, which can make the contract easier to read and maintain. The authors could try to reuse code wherever possible, as this can help reduce the complexity and size of the contract. For instance, the contract could reuse the common code segments in an internal function in order to avoid repeating the same code in multiple places.

IDI - Immutable Declaration Improvement

Criticality	Minor / Informative
Location	CipherEpay.sol#L150
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.

```
maxTotalSupply
```

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.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	CipherEpay.sol#L160,266,280,281,300
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 _taxAddress
uint256 _feeBPS
uint256 _taxBPS
uint256 _deflationBPS
```

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.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	CipherEpay.sol#L138,145,292
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.

```
taxAddress = _taxAddress
initialTokenOwner = tokenOwner
```

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.

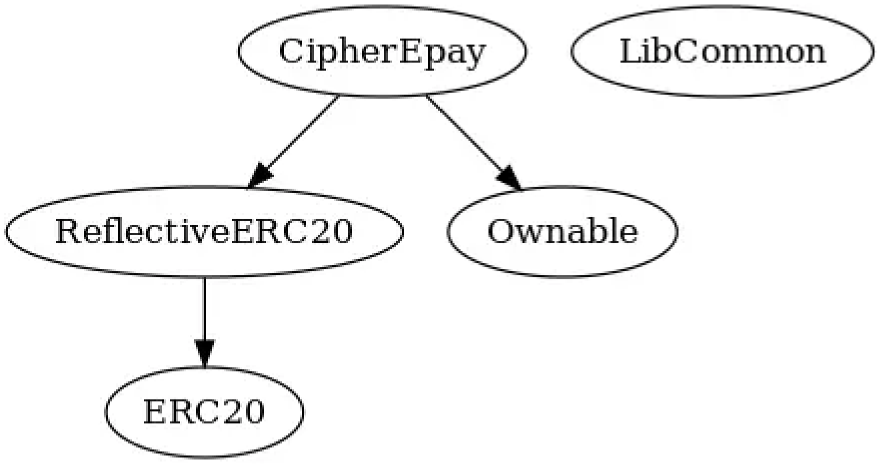
Functions Analysis

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
ReflectiveERC20	Implementation	ERC20		
	_tTotal	Public		-
		Public	✓	ERC20
	balanceOf	Public		-
	transferFrom	Public	✓	-
	transfer	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_setReflectionFee	Internal	✓	
	tokenFromReflection	Public		-
	_transferReflected	Private	✓	
	_reflectFee	Private	✓	
	calculateFee	Private		
	_transferNonReflectedTax	Internal	✓	
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_rUpdate	Private	✓	

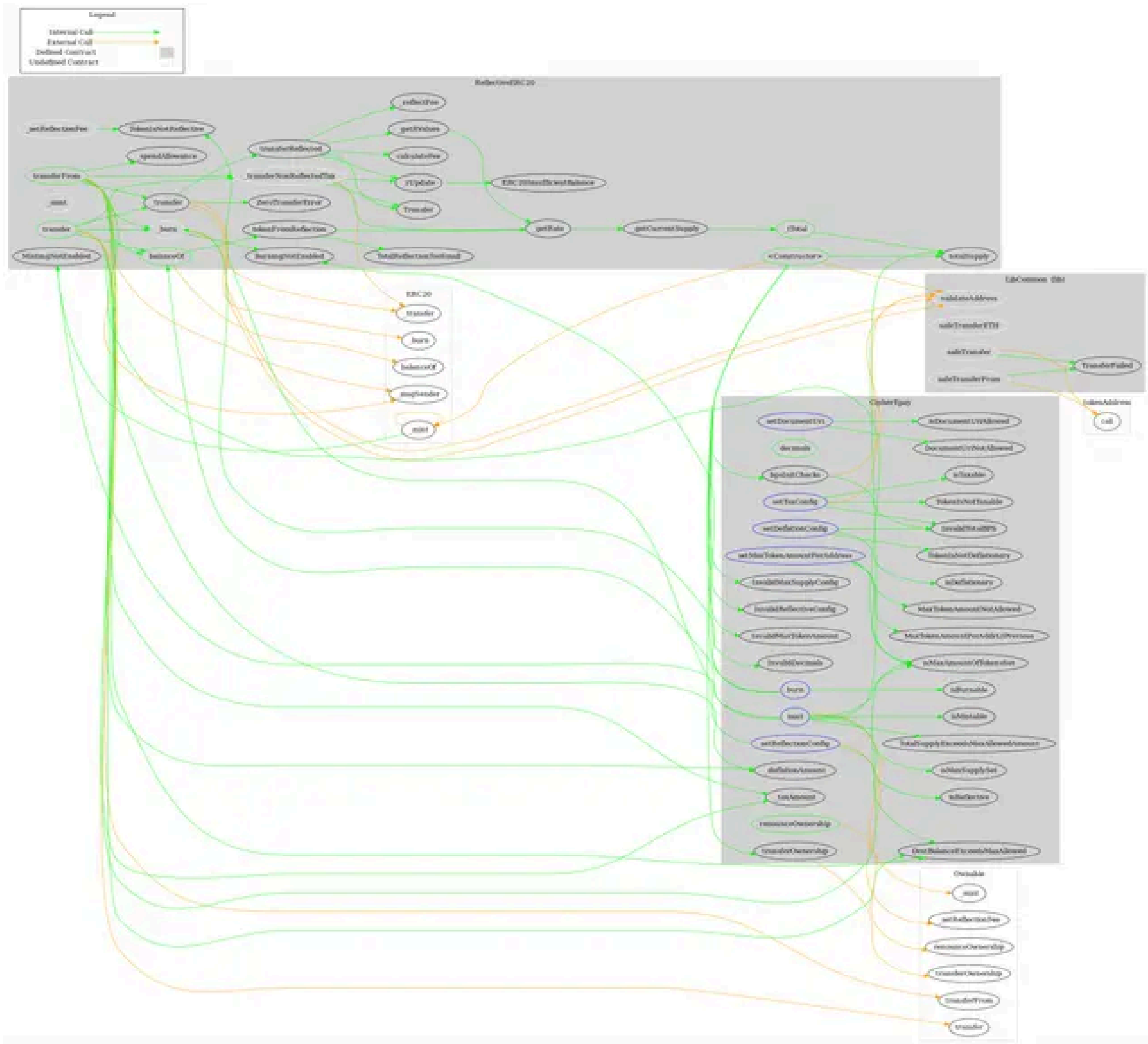
CipherEpay	Implementation	ReflectiveERC20, Ownable		
		Public	✓	ReflectiveERC20
	bpsInitChecks	Private		
	isMintable	Public		-
	isBurnable	Public		-
	isMaxAmountOfTokensSet	Public		-
	isMaxSupplySet	Public		-
	isDocumentUriAllowed	Public		-
	decimals	Public		-
	isTaxable	Public		-
	isDeflationary	Public		-
	isReflective	Public		-
	setDocumentUri	External	✓	onlyOwner
	setMaxTokenAmountPerAddress	External	✓	onlyOwner
	setReflectionConfig	External	✓	onlyOwner
	setTaxConfig	External	✓	onlyOwner
	setDeflationConfig	External	✓	onlyOwner
	transfer	Public	✓	-
	transferFrom	Public	✓	-
	mint	External	✓	onlyOwner
	burn	External	✓	onlyOwner
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_taxAmount	Internal		

	_deflationAmount	Internal		
LibCommon	Library			
	safeTransferETH	Internal	✓	
	validateAddress	Internal		
	safeTransferFrom	Internal	✓	
	safeTransfer	Internal	✓	

Inheritance Graph



Flow Graph



Summary

CIPHER EPAY contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. CIPHER EPAY is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract ownership has been renounced.

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