Cyberscope

Audit Report
Wager

February 2025

SHA256 :

f6ea7bb5be389b46307529c6eb0e35c1d43fd415866548226a093d64fa1297f4

Audited by © cyberscope

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Unresolved
●	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
●	PUPA	Potential Unexcluded Pair Address	Unresolved
●	CCR	Contract Centralization Risk	Unresolved
●	PTRP	Potential Transfer Revert Propagation	Unresolved
●	RC	Redundant Contract	Unresolved
●	RF	Redundant Function	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	Wager
Testing Deploy	https://testnet.bscscan.com/address/0x8d6ddbb55a759e4d86512da03839c85eee067fb1
Symbol	$Wager
Decimals	18
Total Supply	1.000.000

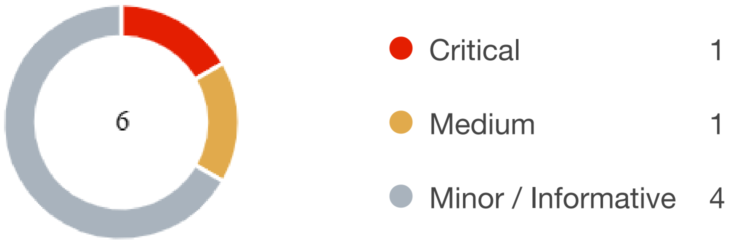
Audit Updates

Initial Audit	12 Feb 2025
Corrected Phase 2	18 Feb 2025

Source Files

Filename	SHA256
Wager.sol	f6ea7bb5be389b46307529c6eb0e35c1d43fd415866548226a093d64fa1297f4

Findings Breakdown



	Severity	Unresolved	Acknowledged	Resolved	Other
●	Critical	1	0	0	0
●	Medium	1	0	0	0
●	Minor / Informative	4	0	0	0

ST - Stops Transactions

Criticality	Critical
Location	Wager.sol#L694,725
Status	Unresolved

Description

The contract owner has the authority to stop the sales for all users excluding the owner. The owner may take advantage of it by setting the `marketingWalletAddress` to a contract address using the `setMarketingWallet` function. As a result, the contract may operate as a honeypot. This is also described in detail in section `PTPR`.

```
if (marketingTokens > 0)
    swapTokensForEth(marketingTokens, marketingWalletAddress);
//...
try

uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
        tokenAmount,
        0,
        path,
        destination,
        block.timestamp
    )
{} catch {
    emit SwapTokensForEthFailed(tokenAmount);
}
```

Recommendation

The team is advised to follow the recommendations outlined in the `PTPR` findings and implement the necessary steps to mitigate the identified risks, ensuring that the contract does not operate as a honeypot. 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.

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.

PUPA - Potential Unexcluded Pair Address

Criticality	Medium
Location	Wager_final_Contract.sol#L505
Status	Unresolved

Description

The contract incorporates operational restrictions on transactions, which can hinder seamless interaction with decentralized applications (dApps) such as launchpads, presales, lockers, or staking platforms. In scenarios where an external contract, such as the pair contract, needs to integrate with the contract, it should be exempt from the limitations to ensure uninterrupted service and functionality. Failure to provide such exemptions can block the successful process and operation of services reliant on this contract.

In this case, while the pair contract is initially excluded from the antibot restrictions, the owner is able to set the mapping `_isExcludedFromAntibot[pair]` to false enabling them again for it.

```
function includeInAntibot(address account) external onlyOwner {
    require(_isExcludedFromAntibot[account], "Account is
already included");
    _isExcludedFromAntibot[account] = false;
    emit IncludeInAntibot(account);
}

function _transfer(**args**) {
    //...
    if (!_isExcludedFromAntibot[from]) {
        require(
            lastTrade[from] + tradeCooldown <= block.number,
            "Trade cooldown not reached"
        );
        lastTrade[from] = block.number;
    }

    if (!_isExcludedFromAntibot[to]) {
        require(
            lastTrade[to] + tradeCooldown <= block.number,
            "Trade cooldown not reached"
        );
        lastTrade[to] = block.number;
    }
    //...
}
```

Recommendation

It is advisable to modify the contract by incorporating functionality that prevents the inclusion of key addresses such as the pair contract in restrictions. This enhancement will allow specific addresses, such as those associated with decentralized applications (dApps) and service platforms, to operate without being hindered by the standard constraints imposed on other users. Implementing this feature will ensure smoother integration and functionality with external systems, thereby expanding the contract's versatility and effectiveness in diverse operational environments.

CCR - Contract Centralization Risk

Criticality	Minor / Informative
Location	Wager_final_Contract.sol#L467,478,486,494,505,513,523,531,539,547,555,563
Status	Unresolved

Description

The contract's functionality and behavior are heavily dependent on external parameters or configurations. While external configuration can offer flexibility, it also poses several centralization risks that warrant attention. Centralization risks arising from the dependence on external configuration include Single Point of Control, Vulnerability to Attacks, Operational Delays, Trust Dependencies, and Decentralization Erosion.

```
function setSwapTokensAtAmount(uint256 amount) external
onlyOwner {/*...*/}
function excludeFromFees(address account) external onlyOwner
{/*...*/}
function includeInFees(address account) external onlyOwner
{/*...*/}
function excludeFromAntibot(address account) external onlyOwner
{/*...*/}
function includeInAntibot(address account) external onlyOwner
{/*...*/}
function excludeMultipleAccountsFromFees(address[] calldata
accounts) external onlyOwner {/*...*/}
function setMarketingWallet(address payable wallet) external
onlyOwner {/*...*/}
function setLiquidityBuyFee(uint256 value) external onlyOwner
{/*...*/}
function setMarketingBuyFee(uint256 value) external onlyOwner
{/*...*/}
function setLiquiditySellFee(uint256 value) external onlyOwner
{/*...*/}
function setMarketingSellFee(uint256 value) external onlyOwner
{/*...*/}
function setTradeCooldown(uint8 newTradeCooldown) external

onlyOwner {/*...*/}
```

Recommendation

To address this finding and mitigate centralization risks, it is recommended to evaluate the feasibility of migrating critical configurations and functionality into the contract's codebase itself. This approach would reduce external dependencies and enhance the contract's self-sufficiency. It is essential to carefully weigh the trade-offs between external configuration flexibility and the risks associated with centralization.

PTRP - Potential Transfer Revert Propagation

Criticality	Minor / Informative
Location	Wager.sol#L694,725
Status	Unresolved

Description

The contract sends funds to a `marketingWalletAddress` as part of the transfer flow. This happens in the `swap` function that calls the router's `swapExactTokensForETHSupportingFeeOnTransferTokens` function. `marketingWalletAddress` can either be a wallet address or a contract. If the address belongs to a contract then it may revert from incoming payment. As a result, the error will propagate to the token's contract and revert the transfer.

`swapExactTokensForETHSupportingFeeOnTransferTokens` uses the `call` method which does not revert and has try/catch to protect the contract from potential reverts but it may drain all the gas used in the transaction reverting the transfer and resulting in the stop of sales.

```
if (marketingTokens > 0)
    swapTokensForEth(marketingTokens, marketingWalletAddress);
//...
try

uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTok
ens(
        tokenAmount,
        0,
        path,
        destination,
        block.timestamp
    )
{} catch {
    emit SwapTokensForEthFailed(tokenAmount);
}
```

Recommendation

The contract should tolerate the potential revert from the underlying contracts when the interaction is part of the main transfer flow. This could be achieved by sending the funds in a non-revertable way. The contract should also tolerate the potential drain of gas by using methods that send a specific amount.

RC - Redundant Contract

Criticality	Minor / Informative
Location	Wager_final_Contract.sol#L750
Status	Unresolved

Description

The `Blacklist` contract has the functionality to declare certain addresses as blacklisted however there is no additional functionality in the contract since `performAction` is empty or in conjunction with other contracts in the provided files. Therefore, the contract is redundant.

```
contract Blacklist {/*...*/}
```

Recommendation

It is recommended to remove code that is not necessary for code optimization and readability.

RF - Redundant Function

Criticality	Minor / Informative
Location	Wager_final_Contract.sol#L781
Status	Unresolved

Description

In the `Blacklist` contract there is a function `performAction` that does not have any functionality.

```
function performAction() public notBlacklisted {
    // Perform contract action
}
```

Recommendation

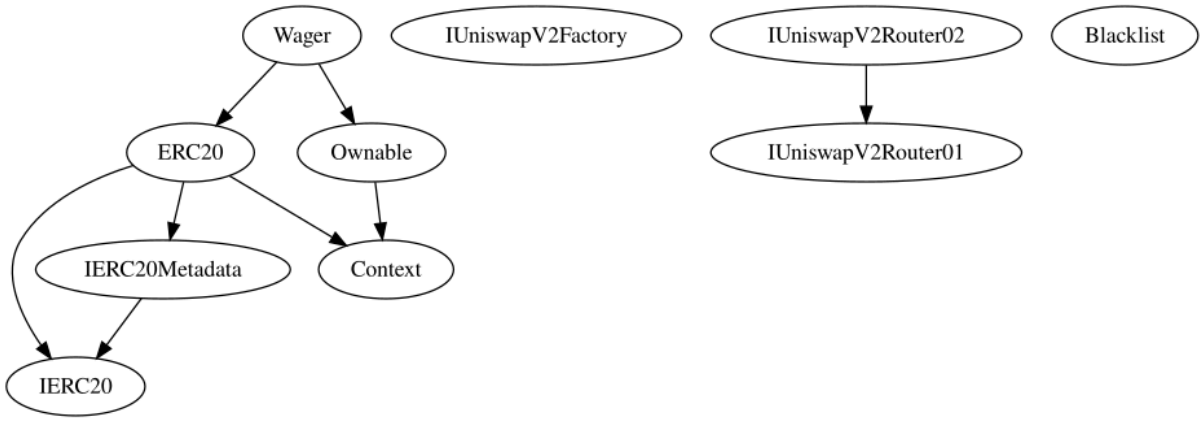
It is recommended to remove empty functions for code optimization and gas cost reduction.

Functions Analysis

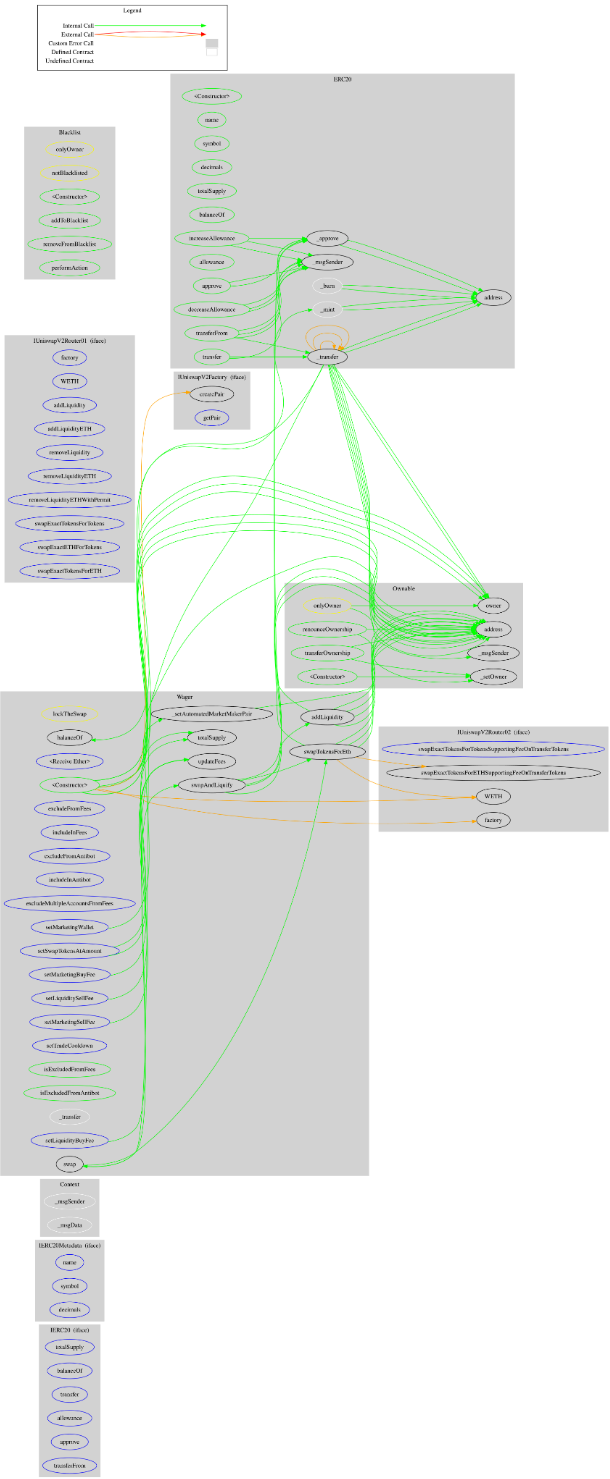
Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Wager	Implementation	ERC20, Ownable		
		Public	✓	ERC20
		External	Payable	-
	setSwapTokensAtAmount	External	✓	onlyOwner
	excludeFromFees	External	✓	onlyOwner
	includeInFees	External	✓	onlyOwner
	excludeFromAntibot	External	✓	onlyOwner
	includeInAntibot	External	✓	onlyOwner
	excludeMultipleAccountsFromFees	External	✓	onlyOwner
	setMarketingWallet	External	✓	onlyOwner
	setLiquidityBuyFee	External	✓	onlyOwner
	setMarketingBuyFee	External	✓	onlyOwner
	setLiquiditySellFee	External	✓	onlyOwner
	setMarketingSellFee	External	✓	onlyOwner
	setTradeCooldown	External	✓	onlyOwner
	updateFees	Internal	✓	
	_setAutomatedMarketMakerPair	Private	✓	
	isExcludedFromFees	Public		-
	isExcludedFromAntibot	Public		-

	_transfer	Internal	✓	
	swap	Private	✓	lockTheSwap
	swapAndLiquify	Private	✓	
	swapTokensForEth	Private	✓	
	addLiquidity	Private	✓	
Blacklist	Implementation			
		Public	✓	-
	addToBlacklist	Public	✓	onlyOwner
	removeFromBlacklist	Public	✓	onlyOwner
	performAction	Public	✓	notBlacklisted

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Custom Error Call
Defined Contract
Undefined Contract
ERC20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
increaseAllowance
allowance
approve
decreaseAllowance
transferFrom
transfer
_approve
_msgSender
_burn
_mint
_transfer
address
Blacklist
onlyOwner
notBlacklisted
<Constructor>
addToBlacklist
removeFromBlacklist
performAction
IUniswapV2Router01 (iface)
factory
WETH
addLiquidity
addLiquidityETH
removeLiquidity
removeLiquidityETH
removeLiquidityETHWithPermit
swapExactTokensForTokens
swapExactETHForTokens
swapExactTokensForETH
IUniswapV2Factory (iface)
createPair
getPair
Ownable
onlyOwner
renounceOwnership
transferOwnership
<Constructor>
owner
address
_msgSender
_setOwner
Wager
lockTheSwap
balanceOf
<Receive Ether>
<Constructor>
excludeFromFees
includeInFees
excludeFromAntibot
includeInAntibot
excludeMultipleAccountsFromFees
setMarketingWallet
setSwapTokensAtAmount
setMarketingBuyFee
setLiquiditySellFee
setMarketingSellFee
setTradeCooldown
isExcludedFromFees
isExcludedFromAntibot
_transfer
setLiquidityBuyFee
swap
_setAutomatedMarketMakerPair
totalSupply
updateFees
swapAndLiquify
addLiquidity
swapTokensForEth
IUniswapV2Router02 (iface)
swapExactTokensForTokensSupportingFeeOnTransferTokens
swapExactTokensForETHSupportingFeeOnTransferTokens
WETH
factory
Context
_msgSender
_msgData
IERC20Metadata (iface)
name
symbol
decimals
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom

Summary

Wager 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 stop transactions. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats. There is also a limit of max 2% fees.

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