Cyberscope

A TAC Security Company

Audit Report

Tikva

January 2026

Network	ETH
Address	0xDb94B53B925262C2CCE845F78DA20e691784f381
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
	MC	Missing Check	Unresolved
	AME	Address Manipulation Exploit	Unresolved
	CCR	Contract Centralization Risk	Unresolved
	OCTD	Transfers Contract's Tokens	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

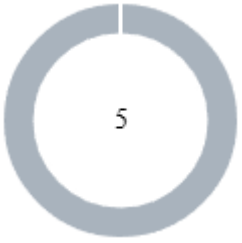
Audit Updates

Initial Audit	26 Jan 2026
Explorer	https://etherscan.io/address/0xDb94B53B925262C2CCE845F78DA20e691784f381
Address	0xDb94B53B925262C2CCE845F78DA20e691784f381
Network	ETHEREUM

Source Files

Filename	SHA256
TikvaPreSale.sol	e4007ed8c7becbc41ab39e0a9e701044bf0598c41d7db31b56dc7d65debed8ad
IERC20.sol	8cdd6ce5124502170e013e0654f13bb5a8cae9847cfe1bf41f34a0d16da07da4

Findings Breakdown



Critical	0
Medium	0
Minor / Informative	5

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	0	0	0
Minor / Informative	5	0	0	0

MC - Missing Check

Criticality	Minor / Informative
Location	TikvaPreSale.sol#L40,59
Status	Unresolved

Description

The contract's `withdraw` `rescueTokens` and functions do not explicitly verify that the contract holds a sufficient token balance before attempting to transfer the requested amount

```
Shell
function withdraw(address to, uint256 amount)
external onlyOwner {
        if (to == address(0)) revert
ZeroAddress();
        if (amount == 0) revert ZeroAmount();


function rescueTokens( address token, address to,
uint256 amount
    ) external onlyOwner {
        if (to == address(0)) revert
ZeroAddress();
        if (amount == 0) revert ZeroAmount();
```

Recommendation

The team is advised to properly check the balances according to the required specifications.

AME - Address Manipulation Exploit

Criticality	Minor / Informative
Location	TikvaPreSale.sol#L67
Status	Unresolved

Description

The contract's design includes functions that accept external contract addresses as parameters without performing adequate validation or authenticity checks. This lack of verification introduces a significant security risk, as input addresses could be controlled by attackers and point to malicious contracts. Such vulnerabilities could enable attackers to exploit these functions, potentially leading to unauthorized actions or the execution of malicious code under the guise of legitimate operations.

```Shell
function rescueTokens( address token, address to,
uint256 amount
    ) external onlyOwner {
        ...
        bool success = IERC20(token).transfer(to,
amount);
```

Recommendation

To mitigate this risk and enhance the contract's security posture, it is imperative to incorporate comprehensive validation mechanisms for any external contract addresses passed as parameters to functions. This could include checks against a whitelist of approved addresses, verification that the address implements a specific contract interface or other methods that confirm the legitimacy and integrity of the external contract. Implementing such validations helps prevent malicious exploits and ensures that only trusted contracts can interact with sensitive functions.

CCR - Contract Centralization Risk

Criticality	Minor / Informative
Location	TikvaPreSale.sol#L40,59,74
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.

```shell
 function withdraw(address to, uint256 amount) external
onlyOwner

 function rescueTokens( address token, address to, uint256
amount ) external onlyOwner


 function rescueETH(address payable to) 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.

OCTD - Transfers Contract's Tokens

Criticality	Minor / Informative
Location	TikvaPreSale.sol#L40,59,74
Status	Unresolved

Description

The contract owner has the authority to claim all the balance of the contract. The owner may take advantage of it by calling the `rescueTokens` , `withdraw` and `rescueETH` functions.

Shell

```
function withdraw(address to, uint256 amount) external
onlyOwner

function rescueTokens(address token, address to, uint256
amount) external onlyOwner


function rescueETH(address payable to) external onlyOwner
```

Recommendation

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.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	TikvaPreSale.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.

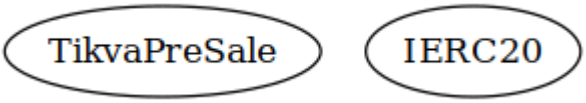
Recommendation

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.

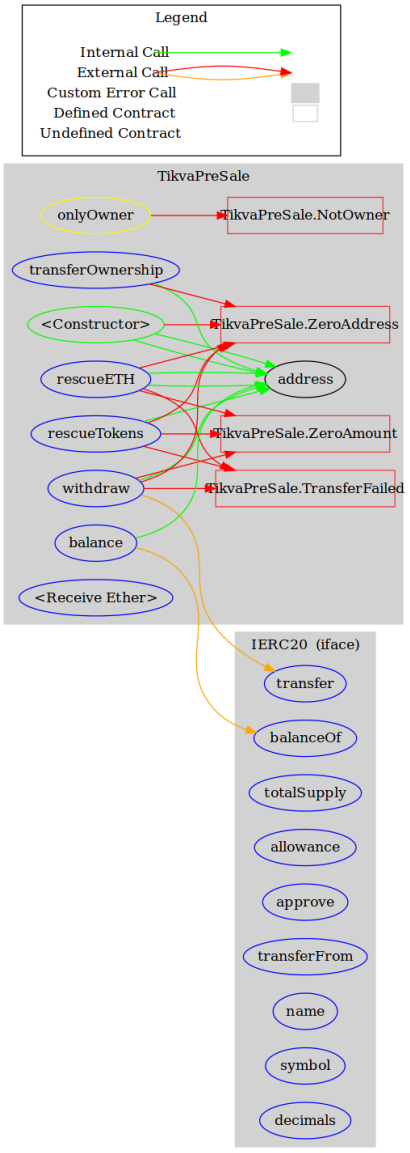
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
TikvaPreSale	Implementation			
		Public	✓	-
	withdraw	External	✓	onlyOwner
	transferOwnership	External	✓	onlyOwner
	rescueTokens	External	✓	onlyOwner
	rescueETH	External	✓	onlyOwner
	balance	External		-
		External	Payable	-
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
	name	External		-

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Custom Error Call
Defined Contract
Undefined Contract
TikvaPreSale
onlyOwner
TikvaPreSale.NotOwner
transferOwnership
<Constructor>
TikvaPreSale.ZeroAddress
rescueETH
address
rescueTokens
TikvaPreSale.ZeroAmount
withdraw
TikvaPreSale.TransferFailed
balance
<Receive Ether>
IERC20 (iface)
transfer
balanceOf
totalSupply
allowance
approve
transferFrom
name
symbol
decimals

Summary

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

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 TAC 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.

A **TAC Security** Company

The Cyberscope team

cyberscope.io