Cyberscope

A TAC Security Company

Audit Report

VaultedMetrics Vesting

March 2026

Network : ETH

Address : 0x88c8b2aeaa1c0701b97e23df36d53d5c1722b49a

Audited by © cyberscope

Table of Contents

Table of Contents **1**
Risk Classification **2**
Review **4**
Audit Updates 4
Source Files 4
Overview **5**
VM888VestingCyberScope.sol 5
Findings Breakdown **6**
Diagnostics **7**
AME - Address Manipulation Exploit 8
Description 8
Recommendation 9
Team Update 9
PTAI - Potential Transfer Amount Inconsistency 10
Description 10
Recommendation 11
Team Update 11
UPR - Unprotected Price Reading 12
Description 12
Recommendation 13
Team Update 13
Functions Analysis **14**
Inheritance Graph **16**
Flow Graph **17**
Summary **18**
Disclaimer **19**
About Cyberscope **20**

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	VM888Vesting
Compiler Version	v0.8.20+commit.a1b79de6
Optimization	200 runs
Explorer	https://etherscan.io/address/0x88c8b2aeaa1c0701b97e23df36d53d5c1722b49a#code
Address	0x88C8b2AEaA1c0701B97E23dF36d53D5c1722B49a
Network	ETHEREUM

Audit Updates

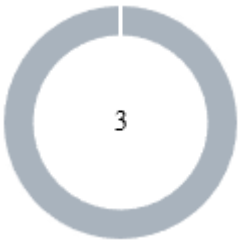
Initial Audit	16 Feb 2026 https://github.com/cyberscope-io/audits/blob/main/vm888/v1/auditVesting.pdf
Corrected Phase 2	03 Mar 2026

Overview

VM888VestingCyberScope.sol

Vaulted Metrics Vesting implements a linear vesting schedule for three hardcoded team allocations totaling 66,600,000 VM888: Founder (70%), VP (20%), and a Future Reserve (10%) whose address is set post-deployment. The contract must be pre-funded before `startVesting()` initializes the vesting clock. Vesting accrues linearly over 5 years from `vestingStart`, with a 12-month cliff preventing any claims during the first year. The `claim()` function transfers the entire claimable balance in a single transaction — partial claims are not supported. Before executing, `claim()` verifies three conditions: the cliff has elapsed, unclaimed vested tokens exist, and the current spot price (read from the token contract) equals or exceeds 120% of the 30-day EMA, preventing team sells during downtrends. The contract uses a custom ownership model with `renounceOwnership()` for full autonomy, and an emergency `recoverERC20()` that explicitly excludes VM888 to protect beneficiary allocations.

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	0	3	0	0

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	AME	Address Manipulation Exploit	Acknowledged
	PTAI	Potential Transfer Amount Inconsistency	Acknowledged
	UPR	Unprotected Price Reading	Acknowledged

AME - Address Manipulation Exploit

Criticality	Minor / Informative
Location	VM888VestingCyberScope.sol#L352
Status	Acknowledged

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 recoverERC20(address tokenAddress, uint256 amount)
external onlyOwner {
    require(tokenAddress != address(vm888), "Cannot recover
VM888");
    (bool success, bytes memory data) = tokenAddress.call(
        abi.encodeWithSelector(0xa9059cbb, msg.sender,
amount)
    );
    require(success && (data.length == 0 ||
abi.decode(data, (bool))), "Transfer failed");

}
```

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.

Team Update

The team has acknowledged that this is not a security issue and states:

The finding is either resolved by our ownership renouncement plan, mitigated by our deployment procedures, or represents an accepted design tradeoff that has been documented internally.

PTAI - Potential Transfer Amount Inconsistency

Criticality	Minor / Informative
Location	VM888VestingCyberScope.sol#L231
Status	Acknowledged

Description

The `claim` function updates the `totalClaimed` state variable with the expected transfer amount. However, the VM888 token implements a fee mechanism on transfers. If the vesting contract is not excluded from tax, the actual amount received by the beneficiary could be less than the recorded amount. As a result, there will be an inconsistency between the tracked claimed amount and the actual amount received by the beneficiary.

```
Shell
totalClaimed[msg.sender] += amount;

if (vm888.balanceOf(address(this)) < amount) revert
InsufficientContractBalance();


bool success = vm888.transfer(msg.sender, amount);
```

Recommendation

The team is advised to ensure that the vesting contract address is excluded from the token's tax mechanism. This will guarantee that the full expected amount is received by the beneficiary during claims, maintaining consistency between the tracked and the actual transferred amount.

Team Update

The team has acknowledged that this is not a security issue and states:

The finding is either resolved by our ownership renouncement plan, mitigated by our deployment procedures, or represents an accepted design tradeoff that has been documented internally.

UPR - Unprotected Price Reading

Criticality	Minor / Informative
Location	VM888VestingCyberScope.sol#L220
Status	Acknowledged

Description

The vesting contract's claim() function verifies that the current spot price exceeds 120% of the 30-day EMA before allowing beneficiaries to withdraw their vested tokens. This price check is intended to prevent team members from claiming during market downtrends. However, both getCurrentPrice() and getEMA() are derived from the Uniswap V2 pair reserves at the moment of the transaction, reflecting instantaneous pool state rather than a time-averaged or manipulation-resistant price. As a result, a beneficiary can temporarily inflate the pool price within a single transaction to satisfy the price condition, bypassing the only protection that prevents token claims during unfavorable market conditions.

```shell
uint256 currentPrice = vm888.getCurrentPrice();
uint256 ema = vm888.getEMA();

if (ema > 0) {
    uint256 requiredPrice = (ema *
PRICE_THRESHOLD_MULTIPLIER) / 1e18;

    if (currentPrice < requiredPrice) revert
PriceConditionNotMet();
```

Recommendation

Price-dependent protocol decisions such as vesting claim gates should not rely on instantaneous spot prices derived from liquidity pool reserves. The team is advised to incorporate a price reading mechanism that is resistant to single-block manipulation, ensuring that the downtrend protection cannot be circumvented through temporary pool state distortion.

Team Update

The team has acknowledged that this is not a security issue and states:

The finding is either resolved by our ownership renouncement plan, mitigated by our deployment procedures, or represents an accepted design tradeoff that has been documented internally.

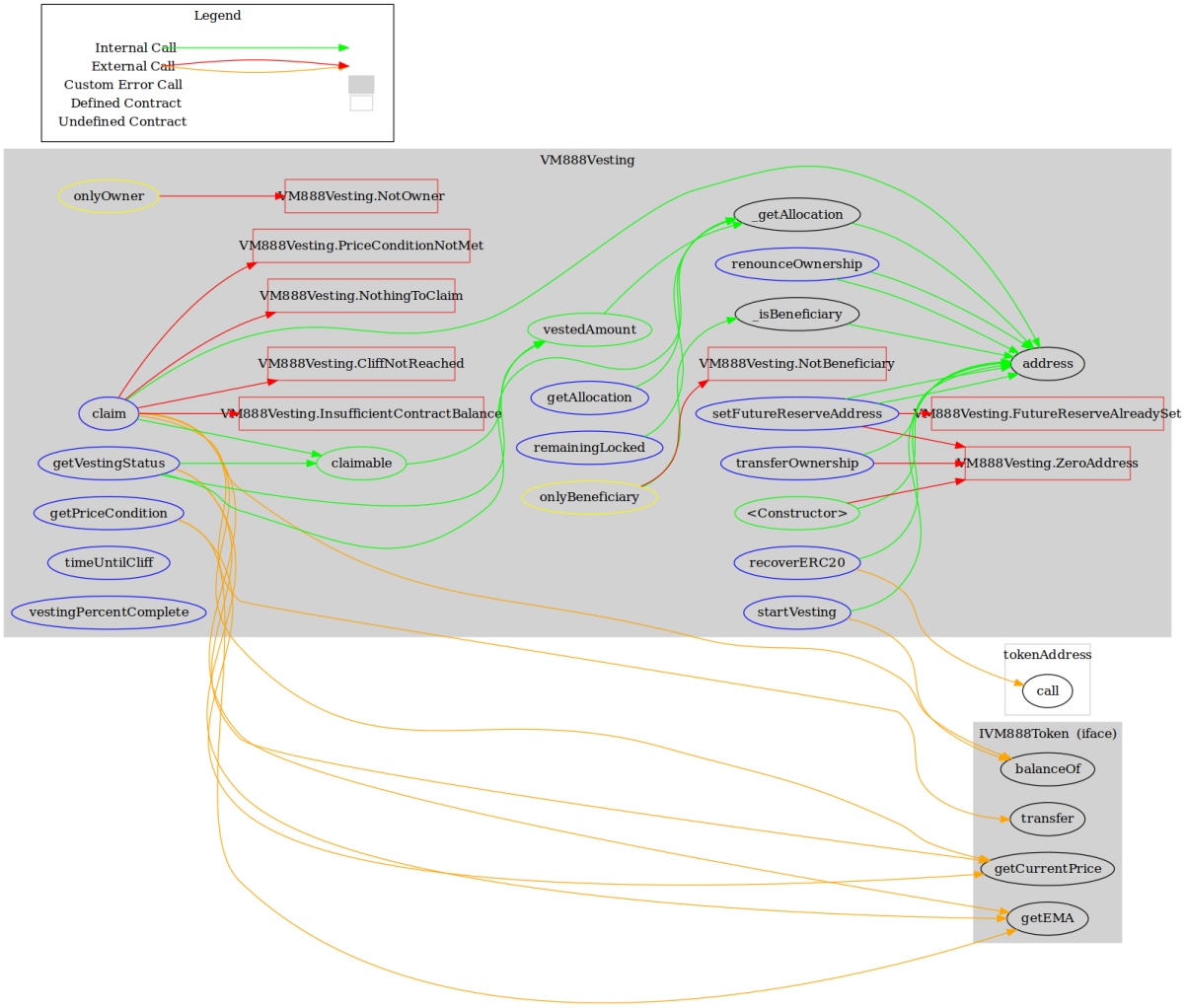
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
VM888Vesting	Implementation			
		Public	✓	-
	startVesting	External	✓	onlyOwner
	vestedAmount	Public		-
	claimable	Public		-
	claim	External	✓	onlyBeneficiary
	getAllocation	External		-
	remainingLocked	External		-
	getPriceCondition	External		-
	getVestingStatus	External		-
	timeUntilCliff	External		-
	vestingPercentComplete	External		-
	setFutureReserveAddress	External	✓	onlyOwner
	transferOwnership	External	✓	onlyOwner
	renounceOwnership	External	✓	onlyOwner
	recoverERC20	External	✓	onlyOwner
	_getAllocation	Private		
	_isBeneficiary	Private		

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Custom Error Call
Defined Contract
Undefined Contract
VM888Vesting
onlyOwner
VM888Vesting.NotOwner
VM888Vesting.PriceConditionNotMet
VM888Vesting.NothingToClaim
VM888Vesting.CliffNotReached
VM888Vesting.InsufficientContractBalance
claim
getVestingStatus
getPriceCondition
timeUntilCliff
vestingPercentComplete
claimable
vestedAmount
getAllocation
remainingLocked
onlyBeneficiary
_getAllocation
renounceOwnership
_isBeneficiary
VM888Vesting.NotBeneficiary
setFutureReserveAddress
transferOwnership
<Constructor>
recoverERC20
startVesting
address
VM888Vesting.FutureReserveAlreadySet
VM888Vesting.ZeroAddress
tokenAddress
call
IVM888Token (iface)
balanceOf
transfer
getCurrentPrice
getEMA

Summary

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