Cyberscope

Audit Report

Dual Pools

May 2023

Network BSC

Audited by © cyberscope

Table of Contents

Review

Repository	https://github.com/JavisJL/dualpools/tree/main/TrendTokenAudit2
Commit	911dc3f722bea6d545f82747f29477bf10327ed3

Deployed Contracts

Contract Name	Explorer
Unitroller	https://bscscan.com/address/0x5E5e28029eF37fC97ffb763C4aC1F532bbD4C7A2
ChainlinkOracle	https://bscscan.com/address/0xCFA47D916Bd512429f05A418d3AF4CA556b03256
VBep20Delegator	https://bscscan.com/address/0x514e2A29e98D49C676c93c5805cb83891CE6a9F5
VBep20Delegator	https://bscscan.com/address/0x3b2A50D0ad420F44f265814029532fCf491201B6
dBUSDDelegator	https://bscscan.com/address/0xB51F589BD9f69a0089c315521EE2FC848bAB6C0c
VBep20Delegator	https://bscscan.com/address/0x5F4a5252880b393a8cc4c01bBA4486Cf7a76075A
dBNB	https://bscscan.com/address/0xB5aAaCcFd69EA45b1A5Aa7E9c7a5e0DB2ce4357e

dUSDT	https://bscscan.com/address/0x514e2A29e98D49C676c93c5805cb83891CE6a9F5
dBUSD	https://bscscan.com/address/0x3b2A50D0ad420F44f265814029532fCf491201B6
dBTCB	https://bscscan.com/address/0xB51F589BD9f69a0089c315521EE2FC848bAB6C0c
dETH	https://bscscan.com/address/0x5F4a5252880b393a8cc4c01bBA4486Cf7a76075A

Audit Updates

Initial Audit	19 Jan 2023 https://github.com/cyberscope-io/audits/blob/main/xdp/v1/audit.pdf
Corrected Phase 2	30 Jan 2023 https://github.com/cyberscope-io/audits/blob/main/xdp/v2/audit.pdf
Corrected Phase 3	02 Feb 2023 https://github.com/cyberscope-io/audits/blob/main/xdp/v3/audit.pdf
Corrected Phase 4	29 May 2023

Source Files

Filename	SHA256
AggregatorV2V3Interface.sol	d1ddf377b603b138396ca9246e6ca0dd3ede629768d9d98c9c44520d1205e585
BEP20Interface.sol	5a126c0688e2a767cf9d14bd5c4bb922c50db92e4e62a622eeefd9dfb36be6fa
CarefulMath.sol	4d7f56d0ff01bb44ff9b6773bf55274574477c816753c22ddd34e658a296f900
Comptroller.sol	dfe96a68b69af5fd52cb91d2eb23fd2c21b8f58c2957b9fa8984f02dcb20e87d
ComptrollerInterface.sol	9bb329b1d7031261da0d207ab6911cfd40fdce0406795868bc15759f42e3465a
ComptrollerLens.sol	29c41591c6504f839e16d6ac5d6822b008cbaf3b1c1752cbdbc70d930cfb9d29
ComptrollerLensInterface.sol	4a02bebdaf14280aa1fce2e6be6f21684479313b9168e7aa070fb624c68f514d
ComptrollerStorage.sol	40e19cbc46daf4b97293b98f1bbdd3ab6120a9115e40db3a79436b384ecad5e5
EIP20Interface.sol	3ee5bbdd464b6b96321cda70c0ee95f4c2676b9292da887814caca9d68da6c81
EIP20NonStandardInterface.sol	03f6818417f9209dc0902f52c2f46227a827a672ad5af0f16b2706e174c09de3
ErrorReporter.sol	4c19c4c0fd78b5077eacf87e917e85c514588432cd70f6cbc37bdfcee8e07d4c
Exponential.sol	00e5b193661b1e003b620461b25565136ea936de83eaf7e6f1e0785a05d5ac27

ExponentialNoError.sol	6700b13c25c4240304a590fda5ab0c1fd5eadf764975e4e6d72ee87ad72643db
InterestRateModel.sol	e7a4beea855785e87adbc63a2e264c44043aa09c5866c94f6766d4ee1388f714
ITradeModel.sol	a8ade90e708124365f8659a6f27826d5ed2cfb3c951358093ca4ac247fd046cf
JumpRateModel.sol	fa0e0eeb6b12a3a34ac2765a507801c9be72529f6c9f7ad705fb5a62c32d3f36
lib.sol	581e167c2bfcdd01484bc09f86f5f8f78c16a2c05525fb23011612bedc6258ce
PriceOracle.sol	fed698ff4b906f82baf23ca905243e01e3a6684363bc329dabf971076b416a5d
SafeMath.sol	4a47d15402f20ec26b0fe15d61f4f6e946e7949b7beaa6398957b5cadee42931
SignedSafeMath.sol	4ba3860fb0de099e2d60dd1f30c2b0342014a0e5a9ed439f1bb68b767f490dd6
TradeModel.sol	ad9771c8b0468ef54aa43dbf1e3b3ff86aad0fd0a7435a1063524b67aef7545b
Unitroller.sol	bb18d95ec5f27d2179deb0d4c9ea8f6ebb02914dec41543a7895462d48c693a0
VAI.sol	1ce1f7718c6a0fe37f100d704aa68f74b353114f7cb038524ebc61b61cd19e50
VAIControllerInterface.sol	ddb382742c00daee01729fb122b57ea48e98474752b7fe414d0b405c81051c1c
VBep20.sol	d64f5412384761c3fad4f9195145088d82d255bc25bc2d669a9c5222871f1f6b
VBep20Delegate.sol	c521f8c21a9124a7929374145363cc0fdbbb36be1ac7211ecda352c043722044

VBep20Delegator.sol	d0cb21874fe8fc98f50d9a338fb8956765413eb34e5dd6250822a729a646dd64
VBep20Immutable.sol	3772c8004f9eda068cc7af805210d9cd704dfddc01ff7224519e6be489f19c32
VBNB.sol	57011e3560b4b8b232cc97e203db1c038eb05581ce546d9edfd24da4199afa1b
VenusChainlinkOracle.sol	30da5aa12f904fb1d0aed035089bfe0fc8c2ca990843fe8f60d17544e77ba3a9
VToken.sol	8280a38a0b53959838ac1fb2eb61ef21b758fdfd0f4528898fec5ac455419fa3
VTokenInterfaces.sol	2ac2ba95a622af014f62dbe668b4d113ff209f6a7a51b8526de06a9ef29cde02
XVS.sol	18d748f021c133ee4fcfa660e973887af8ce782e3a3cfbd9cf4fa84783e4bfc3

Introduction

DualPool implements a mechanism for supplying or borrowing assets. The users submit funds in order to receive vTokens or borrow funds (Cryptocurrency). The submitted funds are operating as collateral. The DualPool also provides a mechanism for trading the supported cryptocurrency with each other. The users have the ability to deposit one cryptocurrency in exchange for another cryptocurrency. The protocol implements a price mechanism that is based on the trade rate of each token.

DualPools is a Venus Protocol fork. This audit focuses on the changes that have been introduced by the DualPools team. The forked project has extended many segments of the Venus codebase. The files that have mainly affected are:

1. Comptroller.sol
2. VToken.sol
3. VBep20.sol
4. VBNB.sol (similar to VBep20.sol)
5. TradeModel.sol

Amount calculation

The DualPool implements a formula to evaluate the price of the underlying tokens based on the trading impact. The price is changed according to the trades similar to a classic DEX logic. According to the whitepaper, this is the price adjustment formula:

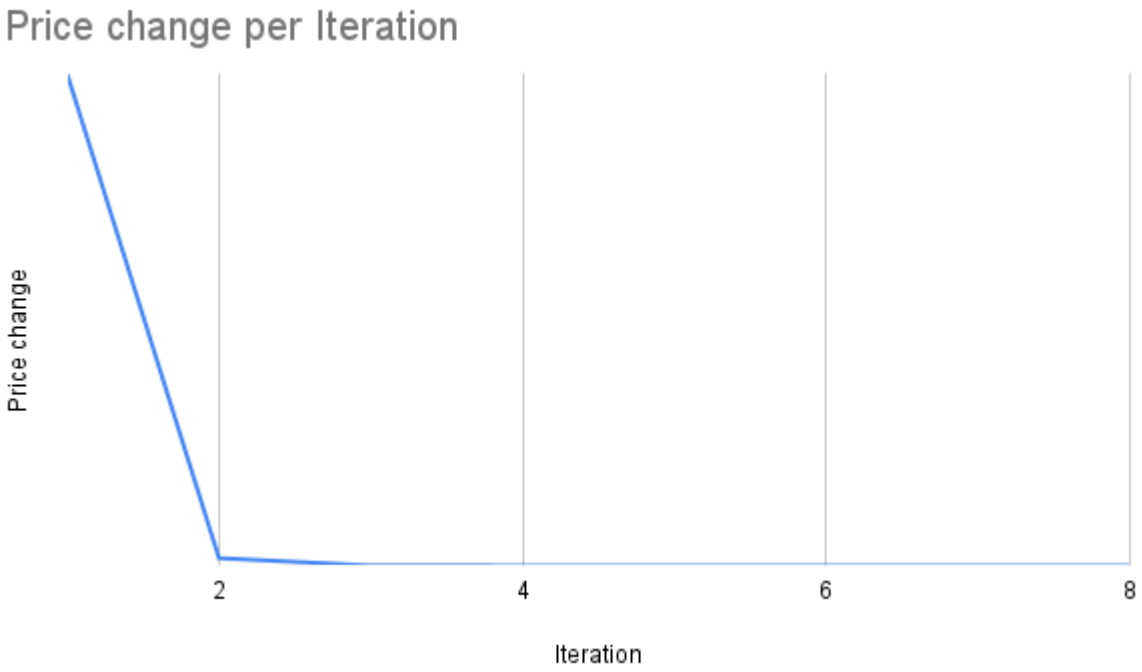
```
iUSDrate = iUSDbalance / (cash*oraclePrice + iUSDbalance)
Price impact = iUSDrate * abs(iUSDrate)
adjustedPrice = oraclePrice * (1 - abs(Price impact))
```

The implementation re-evaluates the adjustedPrice 3 times, providing the new price to the formula on every iteration. The following table depicts the price adjustment re-enforce on every iteration. The calculations are based on the variables

```
iUSDbalance = 1000; Cash = 10000; oraclePrice = 1;
```

Iteration	Price	Change
1	0.9917355372	-
2	0.9916099393	0.0001266445889
3	0.9916080085	0.000001947126855
4	0.9916079788	0.00000002993807002
5	0.9916079783	0.0000000004603129449
6	0.9916079783	0
7	0.9916079783	0
8	0.9916079783	0

We observe that after the third/fourth iteration, the price change tends to zero. Thus it seems a good iteration threshold.



Swap Price Model

The swap feature of the DualPool trades two cryptocurrencies. It accepts one as an exchange for the other. The rate between the two cryptocurrencies depends on two variations.

1. The price of each cryptocurrency.
2. The taxed amount.

As we observe that the well-known decentralized exchange implementation, like Uniswap, the exchange is performed before the price adjustment. Thus, the users are aware of the price that they are going to trade. In the DualPool implementation, the price is adjusted prior to the exchange. We state that this may be the expected behavior of the DualPools business logic, but we mention the diversion with a classic swap mechanism.

https://github.com/Uniswap/v2-core/blob/master/contracts/UniswapV2Pair.sol

Findings Breakdown

Critical	0
Medium	0
Minor / Informative	1

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

Diagnostics

Critical Medium Minor / Informative Pass

Severity	Code	Description	Status
●	PHI	Permissions Handling Inconsistency	Unresolved

PHI - Permissions Handling Inconsistency

Criticality	Minor / Informative
Status	Unresolved

Description

The contract uses admin permissions in order to configure some variables that are essential for the proper operation. The code base contains two different ways of checking the admin permissions. The first one throws a descriptive error message about the failure. The second one has been implemented as a modifier and reverses the execution with a generic authorization message. The diversion of permission handling produced an inconsistency.

```
if (msg.sender != admin) {
    return fail(Error.UNAUTHORIZED,
FailureInfo.SET_PENDING_ADMIN_OWNER_CHECK);
}

modifier onlyAdmin() {
    require(msg.sender == admin,"!admin");
    _;
}
```

Recommendation

The team is advised to introduce one unique permission-handling mechanism. It is recommended to persist in the descriptive message pattern since it is more helpful for the users.

Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
CompDP	Implementation	ComptrollerV8Storage, ComptrollerInterfaceG2, ComptrollerErrorReporter, Exponential NoError		
		Public	✓	-
	ensureAdmin	Private		
	ensureNonzeroAddress	Private		
	getAssetsIn	External		-
	checkMembership	External		-
	enterMarkets	External	✓	-
	addToMarketInternal	Internal	✓	
	exitMarket	External	✓	-
	mintAllowed	External	✓	onlyProtocolAllowed
	mintVerify	External	✓	-
	redeemAllowed	External	✓	onlyProtocolAllowed
	redeemAllowedInternal	Internal		
	redeemVerify	External	✓	-
	borrowAllowed	External	✓	onlyProtocolAllowed

	borrowVerify	External	✓	-
	repayBorrowAllowed	External	✓	onlyProtocolAllowed
	repayBorrowVerify	External	✓	-
	liquidateBorrowAllowed	External	✓	onlyProtocolAllowed
	liquidateBorrowVerify	External	✓	-
	seizeAllowed	External	✓	onlyProtocolAllowed
	seizeVerify	External	✓	-
	transferAllowed	External	✓	onlyProtocolAllowed
	transferVerify	External	✓	-
	getAccountLiquidity	Public		-
	getHypotheticalAccountLiquidity	Public		-
	getHypotheticalAccountLiquidityInternal	Internal		
	liquidateCalculateSeizeTokens	External		-
	liquidateVAICalculateSeizeTokens	External		-
	_setPriceOracle	External	✓	-
	_setCloseFactor	External	✓	-
	_setCollateralFactor	External	✓	-
	_setLiquidationIncentive	External	✓	-
	_setLiquidatorContract	External	✓	-
	_supportMarket	External	✓	-
	_addMarketInternal	Internal	✓	
	_setPauseGuardian	External	✓	-

	_setMarketBorrowCaps	External	✓	-
	_setBorrowCapGuardian	External	✓	-
	_setProtocolPaused	External	✓	validPauseState
	_setVAIController	External	✓	-
	_setVAIMintRate	External	✓	-
	_setTreasuryData	External	✓	-
	_become	External	✓	-
	adminOrInitializing	Internal		
	setVenusSpeedInternal	Internal	✓	
	_setComptrollerLens	External	✓	-
	updateVenusSupplyIndex	Internal	✓	
	updateVenusBorrowIndex	Internal	✓	
	distributeSupplierXDP	Internal	✓	
	distributeBorrowerXDP	Internal	✓	
	claimXDP	Public	✓	-
	claimXDP	Public	✓	-
	claimXDP	Public	✓	-
	claimXDP	Public	✓	-
	grantXDPInternal	Internal	✓	
	_grantXDP	External	✓	-
	_setVenusVAIVaultRate	External	✓	-
	_setVAIVaultInfo	External	✓	-
	_setXDPSpeed	External	✓	-

	getAllMarkets	Public		-
	getBlockNumber	Public		-
	setMintedVAIOf	External	✓	onlyProtocolAllowed
	releaseToVault	Public	✓	-
	getXDPAddress	Public		-
	_pauseTrading	External	✓	-
	dTokenApproved	External		onlyProtocolAllowed
ITradeModel	Interface			
	iUSDrate	External		-
	cashAddUSDMinusLoss	External		-
	newRemoveLiquidityAmt	External		-
	getCashAddUSDMultAbsRate	External		-
	amountsOut	External		-
TradeModel	Implementation	ITradeModel		
		Public	✓	-
	_setTradeFee	External	✓	onlyAdmin
	_setTradeReserveFactor	External	✓	onlyAdmin
	_updateTradeFeeDiscountThresholds	External	✓	onlyAdmin
	_updateTradeFeeDiscountPercents	External	✓	onlyAdmin
	setPriceImpactLimit	External	✓	onlyAdmin
	getValue	Public		-

	getAssetAmt	Public		-
	getValueInt	Public		-
	getAssetAmtInt	Public		-
	abs	Public		-
	iUSDrate	Public		-
	priceImpact	Public		-
	protocolLoss	Public		-
	removeLiquidityFee	Public		-
	newRemoveLiquidityAmt	Public		-
	adjustedPrice	Public		-
	cashAddUSDMinusLoss	Public		-
	getCashAddUSDMultAbsRate	External		-
	feeDiscount	Public		-
	amtAfterFee	Public		-
	amountOutUSDInternal	Public		-
	amountOutTokenInternal	Public		-
	amountsOut	External		-
VBep20	Implementation	VToken, VBep20Interface		
	initialize	Public	✓	-
	mint	External	✓	-
	redeemUnderlying	External	✓	-
	borrow	External	✓	-

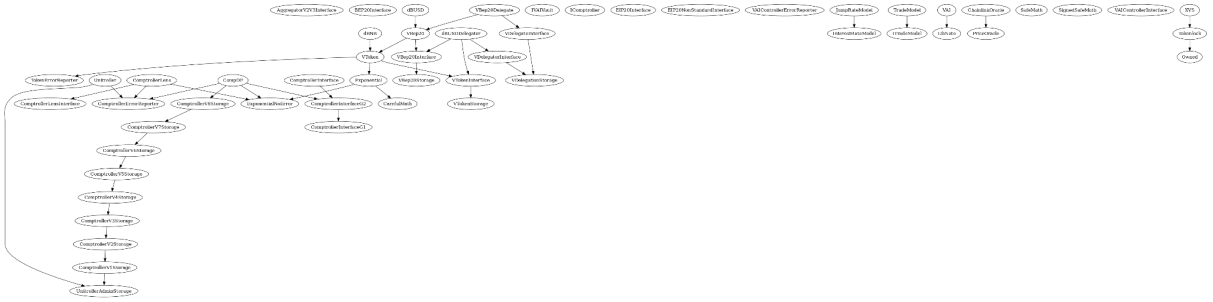
	repayBorrow	External	✓	-
	repayBorrowBehalf	External	✓	-
	liquidateBorrow	External	✓	-
	getCashPrior	Internal		
	doTransferIn	Internal	✓	
	doTransferOut	Internal	✓	
	getCashCurrent	Internal		
	swapExactTokensForTokens	External	✓	nonReentrant
dBNB	Implementation	VToken		
		Public	✓	-
	mint	External	Payable	-
	redeemUnderlying	External	✓	-
	borrow	External	✓	-
	repayBorrow	External	Payable	-
	repayBorrowBehalf	External	Payable	-
	liquidateBorrow	External	Payable	-
		External	Payable	-
	getCashPrior	Internal		
	doTransferIn	Internal	✓	
	doTransferOut	Internal	✓	
	requireNoError	Internal		
	getCashCurrent	Internal		

	swapExactETHForTokens	External	Payable	nonReentrant
VToken	Implementation	VTokenInterface, Exponential, TokenErrorReporter		
	initialize	Public	✓	-
	transferTokens	Internal	✓	
	transfer	External	✓	nonReentrant
	transferFrom	External	✓	nonReentrant
	approve	External	✓	-
	allowance	External		-
	balanceOf	External		-
	balanceOfUnderlying	External	✓	-
	getAccountSnapshot	External		-
	getBlockNumber	Internal		
	borrowRatePerBlock	External		-
	supplyRatePerBlock	External		-
	totalBorrowsCurrent	External	✓	nonReentrant
	borrowBalanceCurrent	External	✓	nonReentrant
	borrowBalanceStored	Public		-
	borrowBalanceStoredInternal	Internal		
	exchangeRateCurrent	Public	✓	nonReentrant
	exchangeRateStored	Public		-
	exchangeRateStoredInternal	Internal		

	getCash	External		-
	accrueInterest	Public	✓	-
	mintInternal	Internal	✓	nonReentrant
	mintFresh	Internal	✓	
	redeemUnderlyingInternal	Internal	✓	nonReentrant
	redeemFresh	Internal	✓	
	borrowInternal	Internal	✓	nonReentrant
	borrowFresh	Internal	✓	
	repayBorrowInternal	Internal	✓	nonReentrant
	repayBorrowBehalfInternal	Internal	✓	nonReentrant
	repayBorrowFresh	Internal	✓	
	liquidateBorrowInternal	Internal	✓	nonReentrant
	liquidateBorrowFresh	Internal	✓	
	seize	External	✓	nonReentrant
	seizeInternal	Internal	✓	
	_setPendingAdmin	External	✓	-
	_acceptAdmin	External	✓	-
	_setComptroller	Public	✓	-
	_setReserveFactor	External	✓	nonReentrant
	_setReserveFactorFresh	Internal	✓	
	_reduceReserves	External	✓	nonReentrant
	_reduceReservesFresh	Internal	✓	
	_setInterestRateModel	Public	✓	-

	_setInterestRateModelFresh	Internal	✓	
	getCashPrior	Internal		
	doTransferIn	Internal	✓	
	doTransferOut	Internal	✓	
	_setLimitIUSD	External	✓	-
	_setTradeModel	External	✓	-
	iUSDrateLimits	Internal		
	subINT	Internal		
	addINT	Internal		
	addUINT	Internal		
	getPriceToken	Public		-
	iUSDrate	Public		-
	removeAmountMinusFee	Public		-
	getExchangeCash	Public		-
	getAvailableCash	Public		-
	amountsOut	Public		-
	getCashCurrent	Internal		
	sendTokenOut	External	✓	nonReentrant

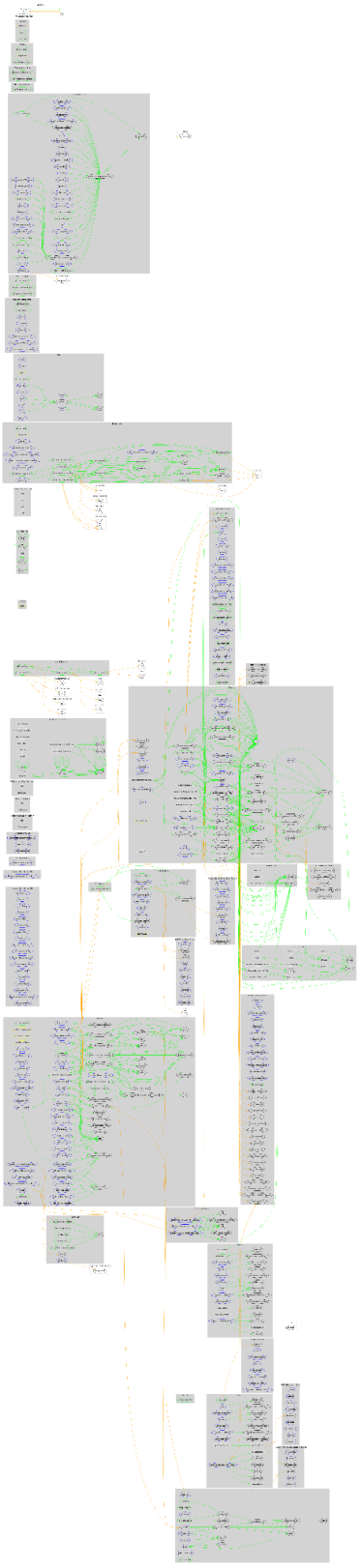
Inheritance Graph



Read the graphs with the original quality on

https://github.com/cyberscope-io/audits/blob/main/xdp

Flow Graph



Summary

Dual Pools contract implements a financial mechanism. This audit investigates security issues, business logic concerns, and potential improvements.

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

https://www.cyberscope.io