GUARDIAN AUDITS

SMART CONTRACT SECURITY AUDIT OF

Summary

Audit Firm Guardian

Prepared By Owen Thurm, Daniel Gelfand, 0xKato, Kiki, Volodya, Omeguhh

Client Firm IVX

Final Report Date September 13th, 2023

Audit Summary

IVX engaged Guardian to review the security of its Options protocol. From the 4th of September to the 12th of September, a team of 6 auditors reviewed the source code in scope. All findings have been recorded in the following report.

Notice that the examined smart contracts are not resistant to internal exploit. For a detailed understanding of risk severity, source code vulnerability, and potential attack vectors, refer to the complete audit report below.

🔗 Blockchain network: **Arbitrum, Avalanche**

✅ Verify the authenticity of this report on Guardian's GitHub: https://github.com/guardianaudits

Table of Contents

Project Overview

Project Summary

Project Name	IVX
Language	Solidity
Codebase	https://github.com/IVX-FI/ivx-diem
Commit(s)	1b77c57207329f3d4c8d619d0c312e292fa73c87

Audit Summary

Delivery Date	September 13, 2023
Audit Methodology	Static Analysis, Manual Review, Test Suite, Contract Fuzzing

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	7	7	0	0	0	0
High	9	9	0	0	0	0
Medium	25	25	0	0	0	0
Low	16	16	0	0	0	0

Audit Scope & Methodology

ID	File	SHA-1 Checksum(s)
DIEM	IVXDiem.sol	db942c5aa99b085613794135eeb40ae30a3101af
DIEMT	IVXDiemToken.sol	268084c885697d4cf4326c94c24ed294ce76c86b
IDIEM	IIVXDiem.sol	49a58431bdc88c63d8663fd7e9b088b9a10a1791
IDIEMT	IIVXDiemToken.sol	7530f37f421d7d8d6330ec346939e23311bceccf
IHEDG	IIVXHedger.sol	49f39d393e20558b4a6b540b0cfa10a08319b8e9
IUNI	IUniswap.sol	fabbf34591126ce2e50a3937acbb441866973e73
IEXNG	IIVXExchange.sol	7cc25dae3073fbbc7bda5310be445c93de1f0773
IAG	IAggregatorV2V3Interface.sol	0b399d13272bd085d8d928d92c76c6c9801cb27e
IDEC	IDecimals.sol	e0b7b79bab844be90ce6df93b57e50c3b7fd77b4
IOCL	IIVXOracle.sol	01c6691a1ce8e8d68f3a28845d8b214dae7e6dd0
IAGG	IAggregatorV3.sol	b838ba0e4ec6971fafc87784defc5fbedc107994
IRSKE	IIVXRiskEngine.sol	3b9d01ad5cc7cc211de7c9ee4f06f92eaeb84264
IPORT	IIVXPortfolio.sol	98e9ea5f64579544dc0a6fdafa7338ce489474d8
ILP	IIVXLP.sol	6a924d14d69006d976fe4cde4e7439eaffae3d59
IQUEUE	IIVXQueue.sol	2f68c118b69c9d4937fbb4daefb2e9ca52b3d4b3
EXNG	IVXExchange.sol	df85faccb34472b2898224b7793a2ca80e2a593a
PXER	IVXPricer.sol	26df5291cfda40fdb8e62a211e084768432bac35
BS	BlackScholes.sol	495a5738855011111e17074ddde41e7ca57298e9
ML	MathLib.sol	ac277054e6f5d4e01c1e27143a5234548c17ec2b
SDM	SignedDecimalMath.sol	c0c77132951864fd88f0a1dd22cebf21688a64fa

Audit Scope & Methodology

ID	File	SHA-1 Checksum(s)
MATH	Math.sol	ab9a7ee819ea4fc9c8f61ad907b772c3bd70de79
DMATH	DecimalMath.sol	e64b8de6f5fb5a145df5c29f15343e36d301d289
THELP	TransferHelper.sol	8687b5e033b19702e20a3154d44e01ffa3e35cb6
CND	ConvertDecimals.sol	f8b7c5d664dd48cb2954fffd8746e6fa05d249da
BIN	Binomial.sol	d4db974fd0443516ae54d7c85a2217b5eb7027da
RSKE	IVXRiskEngine.sol	eefaeaa6496e2a1bcc316b275b8b5915152cbb83
PORT	IVXPortfolio.sol	f4c1732acb6d721cedf826bdf44c5de3442bf530
LP	IVXLP.sol	3450c6ac8db9f44df582a6d86cdb05824ed56853
QUEUE	IVXQueue.sol	898251878daa9faa3f9b4445a792617894cd4d03
OCL	IVXOracle.sol	0766c5c763464145eba9ebb5bfe2454f489c6518

Audit Scope & Methodology

Vulnerability Classifications

Vulnerability Level	Classification
🔴 Critical	Easily exploitable by anyone, causing loss/manipulation of assets or data.
🟠 High	Arduously exploitable by a subset of addresses, causing loss/manipulation of assets or data.
🟡 Medium	Inherent risk of future exploits that may or may not impact the smart contract execution.
🟢 Low	Minor deviation from best practices.

Methodology

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross-referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.
- Comprehensive written tests as a part of a code coverage testing suite.
- Contract fuzzing for increased attack resilience.

Findings & Resolutions

ID	Title	Category	Severity	Status
LP-1	Required Margin Miscalculated	Logical Error	● Critical	Pending
DIEM-1	Liquidations Halted Due To DoS	DoS	● Critical	Pending
DIEM-2	Sell Premium Paid Twice	Logical Error	● Critical	Pending
PORT-1	swapMargin Used To Game The AMM	Griefing	● Critical	Pending
DIEMT-1	DeltaT Rounds Down To 0 Bricking Trades	Rounding	● Critical	Pending
PORT-2	Incorrect Approval Prevents Liquidation	Logical Error	● Critical	Pending
DIEM-3	Liquidation will fail due to insufficient funds	Logical Error	● Critical	Pending
DIEMT-2	vegaDifference Fee Not Valued At The Asset Price	Logical Error	● High	Pending
RSKE-1	Borrowing Fees Accounted For Twice	Logical Error	● High	Pending
LP-2	hedgerTotalLiq Errantly Counted As Debt	Logical Error	● High	Pending
LP-3	Users Can Withdraw Reserved Utilized Collateral	Logical Error	● High	Pending
DIEM-4	Users Can Avoid Borrowing Fees	Logical Error	● High	Pending
DIEM-5	mulDivUp Leads To Unliquidatable Position	Rounding	● High	Pending

Findings & Resolutions

ID	Title	Category	Severity	Status
PORT-3	swapOnUniswap has 0 slippage protection	slippage	High	Pending
RSKE-2	Expired Options Increase Maintenance Margin	Logical Error	High	Pending
PORT-4	Supported Token Removal Gamed	Protocol Manipulation	High	Pending
GLOBAL-1	Unlimited Centralized Controls	Centralization	Medium	Pending
BIN-1	Option May Have Time Value But Zero Premium	Logical Error	Medium	Pending
DIEM-6	PNL Always Decreased By borrowedAmount	Logical Error	Medium	Pending
DIEM-7	Borrowing Fees Extend Past Option Expiry	Unexpected Behavior	Medium	Pending
DIEM-8	averageEntry Always Rounds Up	Rounding	Medium	Pending
DIEMT-3	Errant Fee Applied	Logical Error	Medium	Pending
QUEUE-1	Missing queuedTimestamp Update	Logical Error	Medium	Pending
QUEUE-2	Withdrawal Fees Can Be Gamed	Protocol Manipulation	Medium	Pending
LP-4	utilizationRate May Exceed Max Value	Logical Error	Medium	Pending
RSKE-3	Incorrect X and Y in positionMaintenanceMargin	Logical Error	Medium	Pending

Findings & Resolutions

ID	Title	Category	Severity	Status
DIEM-9	Fees Apply To Insolvent Liquidations	Logical Error	Medium	Pending
GLOBAL-2	Inherent AMM Risk	Protocol Risk	Medium	Pending
OCL-1	Risk-Free Trade by Sandwiching Volatility Updates	Race Condiion	Medium	Pending
DIEM-10	Liquidation Bonus Comes From The Protocol	Logical Error	Medium	Pending
COND-1	Rounding Down Causes Traders Loss	Rounding	Medium	Pending
RSKE-4	Contract can be initialized many times	Access Control	Medium	Pending
GLOBAL-3	Liquidation Will Fail if Oracle is Down	Unexpected Behavior	Medium	Pending
OCL-2	Missing Grace Period Check	Validation	Medium	Pending
DIEMT-4	Alpha Calculation Unused	Unused Feature	Medium	Pending
LP-5	First Depositor Inflation Attack	Protocol Manipulation	Medium	Pending
PORT-5	Liquidation Can Fail Due to Rounding	Rounding	Medium	Pending
EXNG-1	Fixed Pool Can Lead to Bad Swaps	Logical Error	Medium	Pending
DIEM-11	Rounded Funds Stuck in Diem Contract	Logical Error	Medium	Pending

Findings & Resolutions

ID	Title	Category	Severity	Status
QUEUE-3	Malicious User Can Push Deposits	DoS	Medium	Pending
OCL-3	Impossible to Close Option	DoS	Medium	Pending
EXNG-2	No Price Limit on Swaps	Logical Error	Low	Pending
DIEMT-5	ID of 0 is valid for the idModule modifier	Validation	Low	Pending
PORT-6	averageEntry is not being updated on full close trade.	Logical Error	Low	Pending
QUEUE-4	Unnecessary block.timestamp Emitted	Optimization	Low	Pending
OCL-4	Misnamed Variable	Typo	Low	Pending
DIEMT-6	EnumerableSet Should Be Used	Improvement	Low	Pending
QUEUE-5	Superfluous Processed Check	Superfluous Code	Low	Pending
QUEUE-6	Inefficient Use Of Storage Variable	Optimization	Low	Pending
QUEUE-7	Missing Events For reduceQueued Functions	Events	Low	Pending
DIEM-12	Typo	Typo	Low	Pending
DIEM-13	Blacklist Warning	Blacklist	Low	Pending

Findings & Resolutions

ID	Title	Category	Severity	Status
RSKE-5	assetMarginParams Set For An Unsupported Asset	Validation	Low	Pending
DIEMT-7	Parity Check Optimization	Optimization	Low	Pending
GLOBAL-4	Too Many Options Is Not A Good Thing	Warning	Low	Pending
RSKE-6	Inaccurate Variable Names	Naming	Low	Pending
QUEUE-8	Liquidity Amount Does Not Include Fee	Logical Error	Low	Pending

LP-1 | Required Margin Miscalculated

Category	Severity	Location	Status
Logical Error	● Critical	IVXLP.sol: 496, 499	Pending

Description

When computing the position maintenance margin required, the maintenance margin is miscalculated because the `positionMaintenanceMargin` is called with the option premium as the spot price of the asset and the spot price of the asset as the option premium.

This drastically miscalculates the required `positionMaintenanceMargin` and results in an invalid `utilizationRatio`, causing inflated or insufficient interest rates and undermines the `maxUtilizationRatio` validation.

Recommendation

Provide the spot price as the X value and the premium as the Y value.

Resolution

DIEM-1 | Liquidations Halted Due To DoS

Category	Severity	Location	Status
DoS	● Critical	IVXDiem.sol	Pending

Description PoC

There is no limit to the amount of trades a user may open in their portfolio. Therefore it is possible for a user to open so many trades that functions that need to iterate through all of them multiple times, such as `liquidation`, cannot occur.

The `maxBatchTrading` validation fails to protect against this DoS as it limits only the amount of trades that can be opened in a single `openTrades` function call.

Recommendation

Implement a cap on the amount of trades that can belong to any single portfolio.

Resolution

DIEM-2 | Sell Premium Paid Twice

Category	Severity	Location	Status
Logical Error	● Critical	IVXDiem.sol: 374, 452	Pending

Description PoC

In the `openTrades` function, when a user opens a `!isBuy` trade their portfolio receives the `totalPremium` immediately.

However upon settling and closing the expired trade, when the premium of the expired option is 0 the `_trade.averageEntry.mulDivUp(closedUnits, 1e18)`, which represents the `totalPremium` upon opening the sell trade, is credited as PNL in the `_calculatePnl` function and later transferred to the user's portfolio a second time.

Recommendation

Do not credit the initial `totalPremium` as PNL when the sell trade is closed as this amount has already been paid out.

Resolution

PORT-1 | swapMargin Used To Game The AMM

Category	Severity	Location	Status
Griefing	● Critical	IVXPortfolio.sol: 264	Pending

Description PoC

Users are able to use the `swapMargin` function to siphon funds from their portfolio well past the point of insolvency.

- A user can sell options contracts to the AMM and collect the premium in their portfolio immediately.
- The user can then repeatedly use the `swapMargin` function on their portfolio and sandwich themselves to extract roughly all the value from the portfolio.
- The user retains their initial margin deposit as well as the premium from selling the options contracts to the AMM, without having any margin remaining in their portfolio to exercise these contracts upon expiry.

A malicious user can use this attack to drain nearly all of the available funds in the `IVXLP` contract, after removing the balance from their portfolio they will then be insolvently liquidated.

Recommendation

Consider removing the ability to swap margin balances as it poses an inherent risk to the system. Otherwise add validation that the portfolio is not liquidatable at the end of `swapMargin` function:

`if (IIVXDiem(diemContract).isPortfolioLiquidatable(this)) revert IVXPortfolio_PortfolioLiquidatable();`

Resolution

DIEMT-1 | DeltaT Rounds Down To 0 Bricking Trades

Category	Severity	Location	Status
Rounding	● Critical	IVXDiemToken.sol: 260-263	Pending

Description PoC

`secondsToExpiry` is divided by 15 to produce `deltaT`. Options with <15 seconds of expiry will have `deltaT` rounded down to 0 and will cause functions such as `calculateCosts()`, `interestRate()` and `utilizationRatio()` to revert.

The above functions call `Binomial._optionPrices` which calculates N period for binomial pricing:

```
uint256 N = inputs.secondsToExpiry / inputs.deltaT;
```

Because `inputs.deltaT` is 0 after rounding down, a division by 0 revert occurs.
While the option is included in the `activeOptionIds`, all other options will be impacted:

- `interestRate()` and `calculateCosts()` are needed for `pnlOperations`, which will completely prevent closing and liquidation trades from occurring due to the revert. This will negatively impact liquidity providers and traders.
- `calculateCosts()` and `utilizationRatio()` are called when opening a trade, completely preventing trades from being opened in the protocol.

Recommendation

Use the default `inputs.deltaT` value to prevent N from becoming 0. Furthermore, carefully monitor expired options and ensure they are settled to prevent bloating the active options list.

Resolution

PORT-2 | Incorrect Approval Prevents Liquidation

Category	Severity	Location	Status
Logical Error	● Critical	IVXPortfolio.sol: 127	Pending

Description

In the `removeMargin` function, before a swap occurs an amount is approved. This is supposed to be the token amount that is being swapped.

However the `USD` value of the token amount is approved instead. This is especially detrimental when a token's `USD` value is less than $1 because the approval will be insufficient for the swap causing a revert and making liquidations via `removeMargin` impossible.

Recommendation

Approve the token amount that will be swapped instead of the `USD` amount.

Resolution

DIEM-3 | Liquidation will fail due to insufficient funds

Category	Severity	Location	Status
Logical Error	● Critical	IVXDiem.sol: 239	Pending

Description

In the `liquidate` function when a position is insolvently liquidated the `PnL` amount transferred to the `IVXLP` contract is determined to be the total balance of the portfolio in a dollar amount.

If a token in the portfolio is not `USDC`, it will be swapped for `USDC` and incur a fee as well as slippage before the `USDC` amount is received in the `Diem` contract.

Because the Diem contract receives less `USDC`, it will not have sufficient funds to transfer to all necessary stakeholders. The lack of funds will lead to the transaction reverting, making it impossible to liquidate a position that requires a swap.

Recommendation

Use the amount received after `liquidate()` to perform the liquidation during insolvent closes. This will ensure that there are enough funds to finish execution.

Resolution

DIEMT-2 | vegaDifference Fee Not Valued At The Asset Price

Category	Severity	Location	Status
Logical Error	● High	IVXDiemToken.sol: 324, 327	Pending

Description

In the `_calculateFee` function, the `feeTaken` is incremented by the `vegaDifference` factored with the `VEGA_MAKER_FACTOR` or `VEGA_TAKER_FACTOR`.

However the `vegaDifference` is not valued at the underlying asset price with `Oracle.getValuePriced` like the `deltaDifference` is. Therefore the additional fees from the `vegaDifference` are negligible.

Because of this, the effect that the action has on the amm's net vega exposure is not accounted for in the calculated fees and the fees are significantly cheaper than intended.

Recommendation

Calculate the fees from the `vegaDifference` with `Oracle.getValuePriced(vegaDifference.mulDivUp(MakerTakerFactors[_asset].VEGA_MAKER_FACTOR, 1e18), _asset)`.

Resolution

RSKE-1 | Borrowing Fees Accounted For Twice

Category	Severity	Location	Status
Logical Error	● High	IVXRiskEngine.sol: 304	Pending

Description

The borrowing fees are included in the `maintenanceMarginForPositions` which is the numerator for the health factor. However the borrowing fees are also deducted from the PnL of the position in the `calculatePnl` function, therefore reducing the denominator of the health factor.

Therefore the effect of the borrowing fees is doubled when determining the health factor of positions, leading to positions being errantly liquidated.

Recommendation

Adjust the PnL of a trade such that it does not include the borrowing fee, or remove the borrowing fees from the `maintenanceMarginForPositions`.

Resolution

LP-2 | hedgerTotalLiq Errantly Counted As Debt

Category	Severity	Location	Status
Logical Error	High	IVXLP.sol: 477	Pending

Description

The `hedgerTotalLiq` represents the present value of the GMX positions belonging to the protocol. This amount is an asset for liquidity providers in addition to the `NAV` amount held in the contract.

This is in contrast to the other two variables in the numerator of the `utilizationRatio`, the `utilizedCollateral` is a loaned amount and `MM` is a margin maintenance amount.
The `hedgerTotalLiq` is not an amount of the `NAV` that is being utilized, but rather an additional amount of value belonging to the LPs.

Therefore including the `hedgerTotalLiq` in the numerator of the `utilizationRatio` misrepresents the ratio of assets being utilized in the `IVXLP`.

Recommendation

Do not include the `hedgerTotalLiq` in the numerator of the `utilizationRatio`, instead consider factoring it into the `NAV` or including it in the denominator of the `utilizationRatio`.

Resolution

LP-3 | Users Can Withdraw Reserved Utilized Collateral

Category	Severity	Location	Status
Logical Error	● High	IVXLP.sol	Pending

Description

The utilized collateral amount is not factored in when users are depositing and withdrawing, additionally, positions may remain open for a significant amount of time after they are expired. Therefore the utilized collateral can surpass the `NAV`.

As a result, the `totalAvailableAssets` view function will underflow panic revert and the `utilizationRatio` will exceed 100% and perturb the interest rate calculations.

Recommendation

Refactor the option settlement logic such that utilized collateral is reduced to 0 at the end of an epoch when withdrawals are executed.

Resolution

DIEM-4 | Users Can Avoid Borrowing Fees

Category	Severity	Location	Status
Logical Error	● High	IVXDiem.sol	Pending

Description

When users pay borrowing fees, the current interest rate is computed from the `IVXLP` contract and projected across the period from `[trade.timestamp, block.timestamp]`. However the interest rate is variable and will not have been this same value for that entire period.

Users can update their positions when the interest rate drops to lock in the lower rate for the `[trade.timestamp, block.timestamp]` period, even though the interest rate was in fact higher during the majority of that period.

As the interest rate is dependent on the utilized collateral, a malicious actor can wait until another trader closes their `isBuy == true` trade and decreases the interest rate to then update their own trade to lock in the lower rate.
Additionally, a malicious actor could front-run other user's who are closing their trades and increase the interest rate to cause grief.

Recommendation

Refactor the method used to track borrowing fees, such as a per-size approach:
Trades are marked with an initial `borrowingFeePerSize` and upon closing a trade the borrowing fees are computed as the delta between the trade's `latestBorrowingFeePerSize` and the current `borrowingFeePerSize`.

The `borrowingFeePerSize` is updated according to the previous interest rate over the previous period whenever the interest rate is changed.

Resolution

DIEM-5 | mulDivUp Leads To Unliquidatable Position

Category	Severity	Location	Status
Rounding	High	IVXDiem.sol: 411	Pending

Description

In the `_closeTrade` function `mulDivUp` is used to compute the borrowed amount to be subtracted from the utilized collateral. In some cases, when a trade is closed in multiple transactions the resulting decrease of the utilized collateral will be greater than the corresponding increase that opening that trade incurred.

Therefore in some cases the computed `borrowedAmount` to decrease may be greater than the current `utilizedCollateral` amount.

When the computed `borrowedAmount` is greater than the `utilizedCollateral` in the `IVXLP` contract, the transaction will underflow revert and prevent the position from being closed.
Malicious actors can leverage this to halt liquidations and grief other users, preventing positions from being closed.

Recommendation

Refactor the `subUtilizedCollateral` function such that if the provided `_amount` value is greater than the existing `utilizedCollateral` the function does not revert but instead assigns the `utilizedCollateral` to 0.

Resolution

PORT-3 | swapOnUniswap has 0 slippage protection

Category	Severity	Location	Status
slippage	High	IVXPortfolio.sol: 128, 202	Pending

Description

The `removeMargin` and `liquidate` functions have no slippage protection and are therefore vulnerable to sandwich attacks.

These actions can be sandwiched by MEVers to extract a significant amount of value from both users and the AMM.

Recommendation

Allow users to provide a slippage tolerance when they are initiating an action that should have a slippage tolerance.

Upon liquidations, consider refactoring the design such that liquidators are able to seize the assets and transfer them to the IVXLP contract without swapping them. The seized assets can be swapped in a separate transaction.

Resolution

RSKE-2 | Expired Options Increase Maintenance Margin

Category	Severity	Location	Status
Logical Error	High	IVXRiskEngine.sol: 307	Pending

Description

In the `healthFactor` function, the `positionMaintenanceMargin` is added for options which may be expired, however expired options should not increase the required margin for an account — their result is already factored into PnL and that result cannot change.

Recommendation

Skip expired options for the `positionMaintenanceMargin` calculation.

Resolution

PORT-4 | Supported Token Removal Gamed

Category	Severity	Location	Status
Protocol Manipulation	High	IVXPortfolio.sol	Pending

Description

When a token is removed from the `assetsArray` with the `removeAsset` function, it can no longer be used as margin when a portfolio is liquidated.

When the owner calls the `removeAsset` function, a malicious actor can front-run the transaction and make a large trade using the soon-to-be-removed token as margin. After the admin's transaction goes through, the attacker can freely withdraw the now unsupported token through the `withdrawAssets` function.

When the portfolio is attempted to be liquidated, the liquidation will fail as the `PnL` is assigned to the dollar value of the portfolio, which is 0, and the `totalFee` is attempted to be subtracted from the `PnL`.

Recommendation

Only remove supported tokens when options are not tradeable or the protocol is paused.

Resolution

GLOBAL-1 | Unlimited Centralized Controls

Category	Severity	Location	Status
Centralization	Medium	Global	Pending

Description

Throughout the codebase critical values are allowed to be set without appropriate limits on the configured values.

Some of these critical values include:

- collateralFactor
- FEE_TAKEN_PROFITS
- deltaCutoff
- withdrawFee
- limitProcess
- interestRateParams

Recommendation

Implement limits on the range of valid values for each variable.

Resolution

BIN-1 | Option May Have Time Value But Zero Premium

Category	Severity	Location	Status
Logical Error	Medium	Binomial.sol	Pending

Description

An option's premium is calculated to be `Premium = Time Value + Intrinsic Value`.
Consequently, even if an option is out-the-money (no intrinsic value), if there is still time until expiration the premium should be non-zero.

Due to the specific 15 periods and volatility factors used for binomial calculation, it is possible for all of the terminal payoffs to be zero and result in the option's premium to be zero.

- Trader may buy a contract for 0 premium and then sell it as price moves in their favor allowing for nearly risk-free trades.
- Health Ratio Not Validated Correctly
- Put-Call Parity Equation Is Invalidated

Recommendation

Consistently monitor volatility parameters and consider increasing how many periods are used for Binomial options pricing.

Resolution

DIEM-6 | PNL Always Decreased By borrowedAmount

Category	Severity	Location	Status
Logical Error	Medium	IVXDiem.sol: 449	Pending

Description

In the `_calculatePnl` function, for expired buy trades the PNL is decreased by the entire `borrowedAmount` no matter the `closedUnits` provided. This is fine during the `_closeTrade` function call as the `_amountContracts` is set to the entire `trade.contractsOpen`, however other functions that rely on the `_calculatePnl` function do not make this adjustment.

For example the `calculatePnl` function does not make such an adjustment and therefore can give results that may be used to manipulate systems interacting with IVX and relying on the returned value.

Recommendation

Replace `structured.PNL -= int256(_trade.borrowedAmount)` with

`structured.PNL -= int256(_trade.borrowedAmount.mulDivUp(closedUnits, _trade.contractsOpen))`.

Otherwise make the `closedUnits = _trade.contractsOpen` adjustment when the option is expired in the `calculatePnl` `view` function.

Resolution

DIEM-7 | Borrowing Fees Extend Past Option Expiry

Category	Severity	Location	Status
Unexpected Behavior	Medium	IVXDiem.sol: 499	Pending

Description

When a trader closes a trade their borrowing fees are computed based on the period from the `trade.timestamp` to the current `block.timestamp`. However, when an option expires the trader will be closing the trade at a timestamp that is past the expiry time of the option.

Therefore borrowing fees accrue for the option even past it's expiry, when the result has already settled and cannot change. The segment of time between the option expiry and the timestamp of the block in which the trader closes the trade should not be factored when the borrowing fees are computed.

Recommendation

Compute borrowing fees for the period where the option was tradeable and not expired.

Resolution

DIEM-8 | averageEntry Always Rounds Up

Category	Severity	Location	Status
Rounding	Medium	IVXDiem.sol: 336	Pending

Description

In the `_openTrade` function, when the `averageEntry` is computed with an earlier trade, `mulDivUp` is used. However, a higher `averageEntry` is more beneficial for contracts where `isBuy == false`. Malicious traders are therefore able to increase their resulting PNL by splitting their trades up and abusing the round up behavior.

Recommendation

Round up for `isBuy == true` and round down for `isBuy == false`.

Resolution

DIEMT-3 | Errant Fee Applied

Category	Severity	Location	Status
Logical Error	Medium	IVXDiemToken.sol: 371	Pending

Description

The settlement fee is taken on the premium based on the `FEE_TAKEN_PROFITS`, however this calculated amount is not all profit for `isBuy` contracts and is in fact all loss for `!isBuy` contracts.

Recommendation

Do not fee this amount if it represents a loss, additionally compute the fee after the `PNL` has been determined, this way true profits are feed with the `FEE_TAKEN_PROFITS` amount.

Resolution

QUEUE-1 | Missing queuedTimestamp Update

Category	Severity	Location	Status
Logical Error	Medium	IVXQueue.sol: 181	Pending

Description

The `reduceQueuedDeposit` function neglects to update the `queuedTimestamp` of the `queuedDeposit`, however in the `reduceQueuedWithdrawal` function the `queuedTimestamp` is updated.
Any systems relying on this information would be misinformed as the `queuedTimestamp` is not correctly updated.

Recommendation

Update the `queuedTimestamp` in the `reduceQueuedDeposit` function for consistency.

Resolution

QUEUE-2 | Withdrawal Fees Can Be Gamed

Category	Severity	Location	Status
Protocol Manipulation	Medium	IVXQueue.sol: 420	Pending

Description

The accumulated fees from withdrawals during an epoch are distributed to the LP contract after all withdrawals and deposits for that epoch have taken place. This means that the depositors in epoch 10 will receive at least a share of the withdrawal fees from the withdrawals that happened in the same epoch number 10.

This way a profit seeking depositor can observe that many withdrawals are queued for the current epoch and queue a deposit right before the epoch ends to collect these withdrawal fees from individuals who withdrew in the same epoch.

Recommendation

Distribute the withdrawal fees to the depositors who remained in the vault after withdrawals are processed, but before deposits are processed for the current epoch.

Resolution

LP-4 | utilizationRate May Exceed Max Value

Category	Severity	Location	Status
Logical Error	Medium	IVXLP.sol: 449	Pending

Description

It is possible for the `utilizationRatio` to exceed the `maxUtilization`, as the premium value of traded options and the liquidity in hedged positions changes over time.
In this scenario the `utilizationRatio` function will return a ratio greater than the `maxUtilization`, perturbing the interest rate calculations in the `interestRate` function.

Recommendation

Return the `interestRateParams.MaxUtilization` if the `utilizationRatio` exceeds it.

```
 // (utilized collateral + MM + money on gmx) / NAV
+    if(ConvertDecimals.convertTo18(ConvertDecimals.convertFrom18AndRoundUp
+      (_utilizedCollateral + MM + hedgerTotalLiq, _decimals) + .mulDivUp(
+         10 ** _decimals, NAV_Priced), _decimals) >= interestRateParams.MaxUtilization)
+           return interestRateParams.MaxUtilization

return ConvertDecimals.convertTo18( ConvertDecimals.convertFrom18AndRoundUp(_utilizedCollateral + MM +
hedgerTotalLiq, _decimals).mulDivUp( 10 ** _decimals, NAV_Priced ), _decimals );
```

Resolution

RSKE-3 | Incorrect X and Y in positionMaintenanceMargin

Category	Severity	Location	Status
Logical Error	Medium	IVXRiskEngine.sol: 168	Pending

Description

When computing the margin `positionMaintenanceMargin` function, the X and Y values are in an incorrect order.

According to documentation: `Maintenance Margin=a * Max(b * X+c * Y ; d * Y+e * X)`

Recommendation

```
function positionMaintenanceMargin(uint256 X, uint256 Y, address _asset) public view returns (uint256 margin) {
  AssetAttributes memory asset = assetAttributes[_asset];
  margin = asset.marginFactors.marginFactorA.mulDivUp(
    Math.max(
      (asset.marginFactors.marginFactorB * X) + (asset.marginFactors.marginFactorC * Y),
-       (asset.marginFactors.marginFactorD * X) + (asset.marginFactors.marginFactorE * Y) +
+       (asset.marginFactors.marginFactorD * Y) + (asset.marginFactors.marginFactorE * X) ),
    1e36
  );
}
```

Resolution

DIEM-9 | Fees Apply To Insolvent Liquidations

Category	Severity	Location	Status
Logical Error	Medium	IVXDiem.sol: 239, 246	Pending

Description

In cases where a portfolio is insolvently liquidated the fees are still distributed at their original value. This can lead to positions being unable to get liquidated in the event that the `portfolioDollarMargin` is less than the `totalFee`. Though this case may be rare, it is possible with high borrowing fees across many positions and should be handled.

Recommendation

Cap the fees to what is payable in the event of an insolvent liquidation, otherwise consider ignoring them entirely for insolvent liquidations.

Resolution

GLOBAL-2 | Inherent AMM Risk

Category	Severity	Location	Status
Protocol Risk	Medium	Global	Pending

Description

The AMM currently only hedges to be delta neutral. However, it is still vulnerable to volatility risk as the AMM is not vega neutral. Inherently as part of the hedging process, the AMM will have to buy at higher prices and sell at lower prices to maintain delta neutral status. Alongside the volatility in the market, there is a risk that the AMM may not have positive expected value.

Recommendation

Carefully monitor AMM status and increase fees when necessary to protect against losses due to vega non-neutrality.

Resolution

OCL-1 | Risk-Free Trade by Sandwiching Volatility Updates

Category	Severity	Location	Status
Race Condiion	Medium	IVXOracle.sol: 53, 60	Pending

Description

When `setStrikeVolatility()` function is called, it presents an opportunity for an attacker to see that a volatility change will occur and sandwich attack the transaction. In this sandwich attack, the attacker will front-run the volatility change with a buy and then back-run the volatility change with a sell.

By sandwich attacking the transaction, the attacker can profit from the volatility change without exposing themselves to any risk of a price change. This will be a risk-free trade for the attacker at the expense of the LPs.

Recommendation

Increase fees to ensure that the profits from the attack will be less than the fees incurred. Otherwise consider implementing a two-step execution for trading options on the exchange, where a keeper performs the execution of a trade on the behalf of a user.

Resolution

DIEM-10 | Liquidation Bonus Comes From The Protocol

Category	Severity	Location	Status
Logical Error	Medium	IVXDiem.sol: 249	Pending

Description

In the `liquidate()` function when a user's portfolio is liquidated, a `liqBonus` is given to the `msg.sender` who initiates the liquidation.

However the `liqBonus` is subtracted from the `pnl` amount which is to be transferred to the `IVXLP` contract. Instead the `liqBonus` ought to be deducted from the user's remaining portfolio amount if there are leftovers.

Recommendation

Deduct the `liqBonus` from the user's remaining margin amount if it is sufficient rather than deducting it from the amount that the `IVXLP` contract will receive.

Resolution

COND-1 | Rounding Down Causes Traders Loss

Category	Severity	Location	Status
Rounding	Medium	ConvertDecimals.sol: 48	Pending

Description

Once a trader sells an option, their balance is expected to be increased by the option's premium relative to the number of contracts they sold. However, due to the conversion of 18 decimal precision to the precision of the asset, it is possible for a seller to receive no payment for taking on the risk of selling an option.

```
if (assetDecimals < 18) {
// Taking the ceil of 10^(18-decimals) will ensure the first n (asset decimals) have precision when converting
  amount = Math.floor(amount, 10 ** (assetDecimals));
}
```

Any amount below 1 whole unit of an asset will round down to 0, leading to no funds gained on `transferCollateral`.

Recommendation

Enforce a minimum amount of contracts to be traded to avoid rounding issues.

Resolution

RSKE-4 | Contract can be initialized many times

Category	Severity	Location	Status
Access Control	Medium	IVXRiskEngine.sol	Pending

Description

In the `IVXRiskEngine` contract the `initialize` function lacks an `initializer` modifier or any other means to limit the initialization to a single instance.

Therefore the owner may initialize the contract multiple times and change key addresses that otherwise should not change after the `IVXRiskEngine` is in use.

Recommendation

Add validation that the `initialize` function in the `IVXRiskEngine` cannot be called multiple times.

Resolution

GLOBAL-3 | Liquidation Will Fail if Oracle is Down

Category	Severity	Location	Status
Unexpected Behavior	Medium	Global	Pending

Description

In extreme cases, like when oracles go offline or token prices drop to zero, liquidations can get stuck. This poses serious risks to the protocol's financial health.

During these times, it's crucial to allow liquidations to keep the protocol solvent. However, any liquidation-related actions will fail for debt holders of the affected token.
For example, Chainlink has stopped their oracles in rare situations, such as the UST collapse, to avoid giving wrong data to protocols.

If a token's value crashes or the oracle system breaks down, trying to use the `liquidate` function will fail. This is because it depends on the oracle's price information. As a result, users with the affected asset won't face liquidations. This can weaken the protocol's response to solvency issues. There's a risk that a user's asset value could drop below their debts. This would remove any reason to liquidate and push the protocol closer to financial trouble.

Recommendation

Ensure there is a safeguard in place to protect against this possibility. Such as a backup oracle.

Resolution

OCL-2 | Missing Grace Period Check

Category	Severity	Location	Status
Validation	Medium	IVXOracle.sol: 192	Pending

Description

The `getOraclePrice` function lacks grace period validation, therefore if any specific feed is not updated within the grace period a stale price could be used.

Recommendation

Implement a grace period check for the `getOraclePrice` function.

Resolution

DIEMT-4 | Alpha Calculation Unused

Category	Severity	Location	Status
Unused Feature	Medium	IVXDiemToken.sol: 11	Pending

Description

The expiry of an option cannot exceed the end of a queue epoch:

```
//if option expiry bigger than the lp queue next epoch, dont allow creation
if (_option.expiry > IIVXQueue(LP.queueContract()).nextEpochStartTimestamp()) {
  revert CannotCreateOptionWithExpiryAfterNextEpoch();
}
```

Because of this coupling, the expiry of an option is currently limited to 1 day after creation time. This renders any alpha calculation and price blending mechanism useless, as the cutoff of 4 days is never reached. If the blending mechanism were to be used, depositors and withdrawers would have to wait 4 days before depositing/withdrawing funds from the LP.

Recommendation

Consider adjusting the price blending formula and modify the epoch duration appropriately.

Resolution

LP-5 | First Depositor Inflation Attack

Category	Severity	Location	Status
Protocol Manipulation	Medium	IVXLP.sol	Pending

Description

The `IVXLP` vault is susceptible to the first deposit inflation attack.

1) Bob calls `addLiquidity` with 1 wei and then the queue is processed.
2) Bob observes Alice's `addLiquidity` call in the mempool for 100 tokens and frontruns it by transferring 100 tokens directly to the vault to inflate the `NAV`.
3) Once the queue is processed, Alice will mint 0 shares but Bob's 1 share is now worth the entire balance of the vault.

Although this is less likely because only the queue contract can call `mint` and `burn`, the epoch duration is variable and it is a potential risk.

Recommendation

Consider creating “dead” shares by burning some shares on the first deposit or tracking LP balance internally.

Resolution

PORT-5 | Liquidation Can Fail Due to Rounding

Category	Severity	Location	Status
Rounding	Medium	IVXPortfolio.sol: 138	Pending

Description

In the `removeMargin` function, the `amountToRemove` rounds up when being transferred. The issue with this is that by rounding up, it is possible for `amountToRemove` to be greater than the available balance.

This will cause the transfer to revert and potentially prevent liquidations via the `removeMargin` function when the effective margin of the asset is at 100%.

Recommendation

When converting from the price amount to the token amount, do not round up.

Resolution

EXNG-1 | Fixed Pool Can Lead to Bad Swaps

Category	Severity	Location	Status
Logical Error	Medium	IVX.Exchange.sol: 35	Pending

Description

Uniswap pools with the same token pair are differentiated by their configured fee tier. Currently, the `poolFee` is a constant, which means that swaps of that token pair can only occur in the specific pool that has that fee tier.

If liquidity is low in this pool, swaps will occur with a larger price impact than they would otherwise in a different fee tier pool. This will lead to users and the protocol losing funds on swaps unnecessarily.

Recommendation

Allow `poolFee` to be changed so that swaps can happen in the most advantageous pool.

Resolution

DIEM-11 | Rounded Funds Stuck in Diem Contract

Category	Severity	Location	Status
Logical Error	Medium	IVXDiem.sol: 254,255,531,532,533	Pending

Description

The function `convertFrom18AndRoundDown` rounds down by subtracting a value determined by `return x - (x % assetDecimals)`. This deducted amount is then left in the `IVXDiem` contract.

While the amount being locked in the contract is not large, there is no way to access these funds, and each time `convertFrom18AndRoundDown` is called in the `IVXDiem` contract, funds will be locked.

Recommendation

Excess funds that remain after conversions should be claimable by the owner of the portfolio or by the protocol.

Resolution

QUEUE-3 | Malicious User Can Push Deposits

Category	Severity	Location	Status
DoS	Medium	IVXQueue.sol: 161	Pending

Description

A malicious user can prevent the epoch from rolling over by calling `addLiquidity` with multiple 1 wei positions past the `limitProcess`.

This will cause the admin to have to execute multiple transactions to process the queue and expend a potentially significant amount of gas.

Recommendation

Add a minimum deposit amount and consider adding a deposit fee to dissuade these manipulations

Resolution

OCL-3 | Impossible to Close Option

Category	Severity	Location	Status
DoS	Medium	IVXOracle.sol: 191	Pending

Description

During the expiration of an option, it should is finalized using the `settleOptionsExpired` function. Within this function, the `getSpotPriceAtTime` method is invoked, which contains an unbounded loop.

If this loop runs for an extended period, it can exhaust more gas than what's permissible in a single transaction, rendering the option impossible to close. The loop's duration is determined by the number of rounds that transpire between the option's expiration and the invocation of `settleOptionsExpired`.

For tokens with high volatility, the number of rounds can escalate rapidly, potentially leading to a Denial of Service (DoS) situation sooner than anticipated.

Recommendation

Closely monitor the closing of options and ensure the function is called with adequate time before a DoS is possible.

Resolution

EXNG-2 | No Price Limit on Swaps

Category	Severity	Location	Status
Logical Error	● Low	IVXExchange.sol: 98	Pending

Description

There is no price limit set when swaps are performed. With no price limit, a swap can move the price to any amount. This will be especially noticeable when making large trades or when a swap goes through a less liquid pool.

Including traditional slippage protection is a higher priority, but implementing a price limit gives users another way to control slippage.

Recommendation

Consider implementing a price limit that is either set at a fixed percentage of the current price or allow users to choose their own price limit.

Resolution

DIEMT-5 | ID of 0 is valid for the idModule modifier

Category	Severity	Location	Status
Validation	● Low	IVXDiemToken.sol: 54	Pending

Description

The `idModule` modifier allows ids of 0 to pass validation, however an id of 0 is certainly not valid for an option as there are no previous buy/sell call/put combinations.

Impact is limited as all functions utilizing the `idModule` will revert for various reasons upon receiving an id of 0.

Recommendation

Consider specifically disallowing an id of 0 for options in the `idModule` modifier.

Resolution

PORT-6 | averageEntry is not being updated on full close trade.

Category	Severity	Location	Status
Logical Error	Low	IVXPortfolio.sol: L238	Pending

Description

Upon closing a trade with the closeTrade function, the average entry is not assigned to zero. This may result in unexpected behavior for systems relying on this piece of state.

Recommendation

```
function closeTrade(uint256 _optionId) external onlyAllowedContract(diemContract) {
    IIVXDiem.Trade memory traded = optionIdTrade[_optionId];
    traded.timestamp = 0;
+   traded.averageEntry = 0;
    traded.contractsOpen = 0; traded.borrowedAmount = 0;
    optionIdTrade[_optionId] = traded;

    //remove optionId from openOptionIds
    uint256 openOptionIdsLength = openOptionIds.length;
    for (uint256 i; i < openOptionIdsLength; ++i) {
      if (openOptionIds[i] == _optionId) {
          openOptionIds[i] = openOptionIds[openOptionIdsLength - 1];
          openOptionIds.pop();
        break;
      }
    }
}
```

Resolution

QUEUE-4 | Unnecessary block.timestamp Emitted

Category	Severity	Location	Status
Optimization	● Low	IVXQueue.sol: 178, 309	Pending

Description

The `block.timestamp` is unnecessary to emit in the event as it can be retrieved from the block in which the event was emitted.

Recommendation

Remove the timestamp from the `DepositQueued` and `WithdrawQueued` events.

Resolution

OCL-4 | Misnamed Variable

Category	Severity	Location	Status
Typo	Low	IVXOracle.sol: 72	Pending

Description

In the `setValues` function, the parameter of `EncodedData` is labeled as `decodedData`.

Recommendation

Either change the name of the `EncodedData` struct or rename the `decodedData` variable.

Resolution

DIEMT-6 | EnumerableSet Should Be Used

Category	Severity	Location	Status
Improvement	● Low	IVXDiemToken.sol: 141-151	Pending

Description

Throughout the `IVXDiemToken` contract error prone logic is used to add and remove items from the `underlyings` array as if it were a set.

Recommendation

Avoid this error prone logic and use OpenZeppelin's `EnumerableSet` for the `underlyings`.

Resolution

QUEUE-5 | Superfluous Processed Check

Category	Severity	Location	Status
Superfluous Code	● Low	IVXQueue.sol: 131-134	Pending

Description

Checking the `depositsProcessed` and `withdrawalsProcessed` in the `processCurrentQueue` function is superfluous as they can only every be assigned to true together in the `_rolloverEpoch` function where the `currentEpochId` is incremented such that this `epochData` will never be used again in the `processCurrentQueue` function.

Recommendation

Remove the unnecessary `depositsProcessed` and `withdrawalsProcessed` checks.

Resolution

QUEUE-6 | Inefficient Use Of Storage Variable

Category	Severity	Location	Status
Optimization	● Low	IVXQueue.sol: 128	Pending

Description

A stack variable is stored for the `currentEpochId` storage variable, however the `currentEpochId` in storage is still referenced on line 128.

Recommendation

Use the `_currentEpochId` stack variable on line 128.

Resolution

QUEUE-7 | Missing Events For reduceQueued Functions

Category	Severity	Location	Status
Events	● Low	IVXQueue.sol: 181, 312	Pending

Description

The `reduceQueuedWithdrawal` and `reduceQueuedDeposit` functions lack emitted events to signify that the queued action has been reduced.

Recommendation

Implement events for the `reduceQueuedWithdrawal` and `reduceQueuedDeposit` functions.

Resolution

DIEM-12 | Typo

Category	Severity	Location	Status
Typo	Low	IVXDiem.sol: 246	Pending

Description

The comment on line 246 reads `was already transfered to this contract` where transferred is misspelled as `transfered`.

Recommendation

Correct the spelling error.

Resolution

DIEM-13 | Blacklist Warning

Category	Severity	Location	Status
Blacklist	● Low	IVXDiem.sol: 526	Pending

Description

Funds are pushed to the `treasury`, `staker`, and `lp` addresses every time a fee is taken. If any of these addresses are ever blacklisted for the collateral token, the protocol will be DoS'd.

Recommendation

Be aware of this risk and have a contingency plan in place. Otherwise consider refactoring the logic such that funds can be pulled to non-critical addresses such as the treasury.

Resolution

RSKE-5 | assetMarginParams Set For An Unsupported Asset

Category	Severity	Location	Status
Validation	● Low	IVXRiskEngine.sol: 94	Pending

Description

The `changeAssetMarginParams` function can be used to set attributes for an unsupported asset as there is not validation that the provided `_asset` is indeed supported.

Recommendation

Add a requirement that the `supportedAssets[_asset]` entry is `true`.

Resolution

DIEMT-7 | Parity Check Optimization

Category	Severity	Location	Status
Optimization	Low	IVXDiemToken.sol: 132	Pending

Description

In the `createOption` function, `i % 2 == 0` is used to determine the parity of `i`, however `i & 1 == 0` is a more efficient check.

Recommendation

Use `i & 1 == 0` to check the parity of `i`.

Resolution

GLOBAL-4 | Too Many Options Is Not A Good Thing

Category	Severity	Location	Status
Warning	● Low	Global	Pending

Description

The IVX Protocol often relies on enumerating all options contracts and specific trades to compute risk parameters such as the portfolio health factor and LP utilization rate.
However this approach is constrained by the block gas limit and gas expenditure in general.
Therefore limiting the reasonable amount of options and activity that the protocol can support.

Recommendation

Consider re-designing the architecture such that expensive computation does not have to occur for each option contract, and each trade in a user's portfolio. Thereby avoiding `for` loops as much as possible and removing expensive computation from the `for` loops that are necessary.

Resolution

RSKE-6 | Inaccurate Variable Names

Category	Severity	Location	Status
Naming	● Low	IVXRiskEngine.sol: 277-287	Pending

Description

`Vega_shockLoss` is set to the delta shock value and `Delta_shockLoss` is set to the vega shock value.

```
if (SumDeltaShock_negative < SumDeltaShock_positive) {
    Vega_shockLoss = SumDeltaShock_negative;
} else {
    Vega_shockLoss = SumDeltaShock_positive;
}
if (SumVegaShock_negative < SumVegaShock_positive) {
    Delta_shockLoss = SumVegaShock_negative;
} else {
    Delta_shockLoss = SumVegaShock_positive;
}
```

Recommendation

Switch the variable names to accurately reflect the values they represent.

Resolution

QUEUE-8 | Liquidity Amount Does Not Include Fee

Category	Severity	Location	Status
Logical Error	● Low	IVXQueue.sol: 374	Pending

Description

`LP.withdrawLiquidity` returns the amount of liquidity withdrawn without accounting for the withdrawal fee.

This amount is then stored in the mapping `depositEpochQueue` and does not accurately reflect how much liquidity was returned to the depositor.

Recommendation

Consider whether the amount after the fee is necessary, and if so set `_amount` to `_amount-_fee`.

Resolution

Disclaimer

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 considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Guardian to perform a security assessment. This report 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 in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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. Guardian’s position is that each company and individual are responsible for their own due diligence and continuous security. Guardian’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 Guardian is subject to dependencies and 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.

Notice that smart contracts deployed on the blockchain are not resistant from internal/external exploit. Notice that active smart contract owner privileges constitute an elevated impact to any smart contract’s safety and security. Therefore, Guardian does not guarantee the explicit security of the audited smart contract, regardless of the verdict.

About Guardian Audits

Founded in 2022 by DeFi experts, Guardian Audits is a leading audit firm in the DeFi smart contract space. With every audit report, Guardian Audits upholds best-in-class security while achieving our mission to relentlessly secure DeFi.

To learn more, visit https://guardianaudits.com

To view our audit portfolio, visit https://github.com/guardianaudits

To book an audit, message https://t.me/guardianaudits