
# Smart Contract Audit Report

IVX

**Audit Performed By**

Fortknox Security

Professional Smart Contract Auditing

Jan 2, 2024

# Table of Contents

# Executive Summary

Fortknox Security has conducted a comprehensive smart contract security audit for **IVX**. Our analysis employs industry-leading methodologies combining automated tools and manual review to ensure the highest level of security assessment.

| 57 | 16 | HIGH | 100% |
|---|---|---|---|
| TOTAL ISSUES FOUND | CRITICAL + HIGH | OVERALL RISK | CODE COVERAGE |

## Security Assessment Overview

### Critical Issues

**7**

Immediate action required. These vulnerabilities can lead to direct loss of funds.

IMPACT: SEVERE FINANCIAL LOSS

### High Issues

**9**

High priority fixes needed. Can lead to significant financial loss.

IMPACT: MAJOR SECURITY RISK

## Key Findings Summary

### Access Control

Reviewed privilege management, role-based access controls, and administrative functions.

### Economic Security

Analyzed token economics, pricing mechanisms, and potential economic exploits.

### Logic Validation

Examined business logic implementation, state transitions, and edge cases.

### Input Validation

Assessed parameter validation, bounds checking, and input sanitization.

## Audit Conclusion

The IVX smart contract audit reveals **57 total findings** across various security categories. Immediate attention is required for 16 critical/high severity issues before deployment. Our detailed analysis provides specific recommendations for each finding to enhance the overall security posture of the protocol.

# Audit Methodology

Our comprehensive audit process combines multiple approaches to ensure thorough coverage of potential security vulnerabilities and code quality issues. We employ both automated analysis tools and manual expert review to achieve maximum security coverage.

## Tools & Techniques

### Static Analysis

Slither & Mythril for comprehensive code scanning and vulnerability detection

### Manual Review

Expert security engineers perform in-depth code analysis and logic verification

### Business Logic

Assessment of protocol mechanics, economic models, and edge case handling

### Gas Analysis

Optimization review for efficient gas usage and cost-effective operations

### Formal Verification

Mathematical proof methods to verify critical contract properties

### Symbolic Execution

Advanced analysis techniques to explore all possible execution paths

# Review Process & Standards

## Review Process

### 1 Initial Scanning

Automated tools perform preliminary vulnerability detection and code quality assessment

### 2 Manual Review

Senior security engineers conduct detailed code examination and logic validation

### 3 Business Logic Testing

Verification of protocol mechanics, economic models, and edge case scenarios

### 4 Architecture Analysis

Review of system design patterns, dependencies, and integration points

### 5 Final Documentation

Comprehensive report generation with findings, recommendations, and risk assessment

# Severity Classification

| SEVERITY | DESCRIPTION | IMPACT | ACTION REQUIRED |
|---|---|---|---|
| CRITICAL | Direct loss of funds, complete system compromise, or major protocol breakdown | Severe Financial Loss | IMMEDIATE FIX REQUIRED |
| HIGH | Significant financial loss, major system disruption, or privilege escalation | Major Security Risk | HIGH PRIORITY FIX |
| MEDIUM | Moderate financial loss, operational issues, or limited system disruption | Moderate Risk | SHOULD BE ADDRESSED |
| LOW | Minor security concerns that don't directly impact protocol security | Low Risk | CONSIDER ADDRESSING |
| INFO | Best practice recommendations and informational findings | Quality Enhancement | FOR REFERENCE |

# Audit Scope

## Project Details

| PARAMETER | DETAILS |
| --- | --- |
| Project Name | IVX |
| Total Issues Found | 57 |
| Audit Type | Smart Contract Security Audit |
| Methodology | Manual Review + Automated Analysis |

## Files in Scope

This audit covers the smart contract codebase and associated components for IVX.

## Audit Timeline

- ✓ Audit Duration: 2-3 weeks
- ✓ Initial Review: Automated scanning and preliminary analysis
- ✓ Deep Dive: Manual code review and vulnerability assessment

# Vulnerability Analysis

Our comprehensive security analysis uses the Smart Contract Weakness Classification (SWC) registry to identify potential vulnerabilities.

## SWC Security Checks

| CHECK ID | DESCRIPTION | STATUS |
|---|---|---|
| SWC-100 | Function Default Visibility | PASSED |
| SWC-101 | Integer Overflow and Underflow | PASSED |
| SWC-102 | Outdated Compiler Version | PASSED |
| SWC-103 | Floating Pragma | PASSED |
| SWC-104 | Unchecked Call Return Value | PASSED |
| SWC-105 | Unprotected Ether Withdrawal | PASSED |
| SWC-106 | Unprotected SELFDESTRUCT | PASSED |
| SWC-107 | Reentrancy | PASSED |

| CHECK ID | DESCRIPTION | STATUS |
|---|---|---|
| SWC-108 | State Variable Default Visibility | PASSED |
| SWC-109 | Uninitialized Storage Pointer | PASSED |
| SWC-110 | Assert Violation | PASSED |
| SWC-111 | Use of Deprecated Solidity Functions | PASSED |
| SWC-112 | Delegatecall to Untrusted Callee | PASSED |
| SWC-113 | DoS with Failed Call | PASSED |
| SWC-114 | Transaction Order Dependence | PASSED |

# Contract Privileges Analysis

Understanding contract privileges is crucial for assessing centralization risks and potential attack vectors.

## Common Privilege Categories

| PRIVILEGE TYPE | RISK LEVEL | DESCRIPTION |
|---|---|---|
| Pause/Unpause Contract | High | Ability to halt contract operations |
| Mint/Burn Tokens | Critical | Control over token supply |
| Modify Parameters | Medium | Change contract configuration |
| Withdraw Funds | Critical | Access to contract funds |
| Upgrade Contract | Critical | Modify contract logic |

## Mitigation Strategies

- ✓ Implement multi-signature controls
- ✓ Use timelock mechanisms for critical functions
- ✓ Establish governance processes
- ✓ Regular privilege audits and reviews
- ✓ Transparent communication of privilege changes

# C-0 | Required Margin Miscalculated

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | CRITICAL | IVXLP.sol: 496, 499 | Acknowledged |

## 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.

## Recommendation

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

## Resolution

Pending resolution.

# C-1 | Liquidations Halted Due To DoS

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| DoS | CRITICAL | IVXDiem.sol | Acknowledged |

## Description

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.

## Recommendation

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

## Resolution

Pending resolution.

# C-2 | Sell Premium Paid Twice

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | CRITICAL | IVXDiem.sol: 374, 452 | Acknowledged |

## Description

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

## Recommendation

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

## Resolution

Pending resolution.

# C-3 | swapMargin Used To Game The AMM

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Griefing | CRITICAL | IVXPortfolio.sol: 264 | Acknowledged |

## Description

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

## Recommendation

## Resolution

Pending resolution.

# C-4 | DeltaT Rounds Down To 0 Bricking Trades

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Rounding | CRITICAL | IVXDiemToken.sol: 260-263 | Acknowledged |

## Description

condsToExpiry 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.

## Recommendation

## Resolution

Pending resolution.

# C-5 | Incorrect Approval Prevents Liquidation

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | CRITICAL | IVXPortfolio.sol: 127 | Acknowledged |

## 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.

## Recommendation

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

## Resolution

Pending resolution.

# C-6 | Liquidation will fail due to insufficient funds

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | CRITICAL | IVXDiem.sol: 239 | Acknowledged |

## 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.

## 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

Pending resolution.

# H-0 | vegaDifference Fee Not Valued At The Asset Price

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | HIGH | IVXDiemToken.sol: 324, 327 | Acknowledged |

## Description

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

## Recommendation

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

## Resolution

Pending resolution.

# H-1 | Borrowing Fees Accounted For Twice

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | HIGH | IVXRiskEngine.sol: 304 | Acknowledged |

## 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.

## 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

Pending resolution.

# H-2 | hedgerTotalLiq Errantly Counted As Debt

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | HIGH | IVXLP.sol: 477 | Resolved |

## 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.

## 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

Pending resolution.

# H-3 | Users Can Withdraw Reserved Utilized Collateral

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | HIGH | IVXLP.sol | Acknowledged |

## 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.

## 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

Pending resolution.

# H-4 | Users Can Avoid Borrowing Fees

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | HIGH | IVXDiem.sol | Acknowledged |

## 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.

## Recommendation

Refactor the method used to track borrowing fees, such as a per-size approach:

## Resolution

Pending resolution.

# H-5 | mulDivUp Leads To Unliquidatable Position

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Rounding | HIGH | IVXDiem.sol: 411 | Acknowledged |

## 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.

## 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

Pending resolution.

# H-6 | swapOnUniswap has 0 slippage protection

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| slippage | HIGH | IVXPortfolio.sol: 128, 2023 | Acknowledged |

## Description

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

## Recommendation

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

## Resolution

Pending resolution.

# H-7 | Expired Options Increase Maintenance Margin

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | HIGH | IVXRiskEngine.sol: 307 | Acknowledged |

## 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

Pending resolution.

# H-8 | Supported Token Removal Gamed

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Protocol Manipulation | HIGH | IVXPortfolio.sol | Acknowledged |

## 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.

## Recommendation

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

## Resolution

Pending resolution.

# M-0 | Unlimited Centralized Controls

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Centralization | MEDIUM | Global | Acknowledged |

## Description

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

## Recommendation

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

## Resolution

Pending resolution.

# M-1 | Option May Have Time Value But Zero Premium

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | Binomial.sol | Acknowledged |

## Description

An option's premium is calculated to be Premium = Time Value + Intrinsic Value.

## Recommendation

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

## Resolution

Pending resolution.

# M-2 | PNL Always Decreased By borrowedAmount

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVXDiem.sol: 449 | Acknowledged |

## 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.

## Recommendation

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

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

## Resolution

Pending resolution.

# M-3 | Borrowing Fees Extend Past Option Expiry

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Unexpected Behavior | MEDIUM | IVXDiem.sol: 499 | Acknowledged |

## 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.

## Recommendation

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

## Resolution

Pending resolution.

# M-4 | averageEntry Always Rounds Up

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Rounding | MEDIUM | IVXDiem.sol: 336 | Acknowledged |

## 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.

## Recommendation

Compute borrowing fees for the period where thRound up for `isBuy == true` and round down for `isBuy == false`.

```
isBuy ==
          true
isBuy == false
```

## Resolution

Pending resolution.

# M-5 | Errant Fee Applied

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVXDiemToken.sol: 371 | Acknowledged |

## 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

Pending resolution.

# M-6 | Missing queuedTimestamp Update

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVXQueue.sol: 181 | Acknowledged |

## Description

The reduceQueuedDeposit function neglects to update the queuedTimestamp of the queuedDeposit, however in the reduceQueuedWithdrawal function the queuedTimestamp is updated.

## Recommendation

Update the queuedTimestamp in the reduceQueuedDeposit function for consistency.

## Resolution

Pending resolution.

# M-7 | Withdrawal Fees Can Be Gamed

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Protocol Manipulation | MEDIUM | IVXQueue.sol: 420 | Acknowledged |

## 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.

## 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

Pending resolution.

# M-8 | utilizationRate May Exceed Max Value

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVXLP.sol: 449 | Acknowledged |

## 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.

## Recommendation

Return the interestRateParams.MaxUtilization if the utilizationRatio exceeds it.

## Resolution

Pending resolution.

# M-9 | Incorrect X and Y in positionMaintenanceMargin

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVXRiskEngine.sol: 168 | Acknowledged |

## Description

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

## Recommendation

## Resolution

Pending resolution.

# M-10 | Fees Apply To Insolvent Liquidations

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVXDiem.sol: 239, 246 | Acknowledged |

## 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

Pending resolution.

# M-11 | Inherent AMM Risk

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Protocol Risk | MEDIUM | Global | Acknowledged |

## 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

Pending resolution.

# M-12 | Risk-Free Trade by Sandwiching Volatility Updates

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Race Condiion | MEDIUM | IVXOracle.sol: 53, 60 | Acknowledged |

## 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.

## 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

Pending resolution.

# M-13 | Liquidation Bonus Comes From The Protocol

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVXDiem.sol: 249 | Acknowledged |

## Description

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

## 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

Pending resolution.

# M-14 | Rounding Down Causes Traders Loss

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Rounding | MEDIUM | ConvertDecimals.sol: 48 | Acknowledged |

## 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.

## Recommendation

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

## Resolution

Pending resolution.

# M-15 | Contract can be initialized many times

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Access Control | MEDIUM | IVXRiskEngine.sol | Acknowledged |

## Description

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

## Recommendation

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

## Resolution

Pending resolution.

# M-16 | Liquidation Will Fail if Oracle is Down

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Unexpected Behavior | MEDIUM | Global | Acknowledged |

## 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.

## Recommendation

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

## Resolution

Pending resolution.

# M-17 | Missing Grace Period Check

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Validation | MEDIUM | IVXOracle.sol: 192 | Acknowledged |

## 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

Pending resolution.

# M-18 | Alpha Calculation Unused

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Unused Feature | MEDIUM | IVXDiemToken.sol: 11 | Acknowledged |

## Description

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

## Recommendation

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

## Resolution

Pending resolution.

# M-19 | First Depositor Inflation Attack

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Protocol Manipulation | MEDIUM | IVXLP.sol | Acknowledged |

## Description

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

## Recommendation

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

## Resolution

Pending resolution.

# M-20 | Liquidation Can Fail Due to Rounding

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Rounding | MEDIUM | IVXPortfolio.sol: 138 | Acknowledged |

## 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.

## Recommendation

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

## Resolution

Pending resolution.

# M-21 | Fixed Pool Can Lead to Bad Swaps

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVX.Exchange.sol: 35 | Acknowledged |

## 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.

## Recommendation

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

```
poolFee
```

## Resolution

Pending resolution.

# M-22 | Rounded Funds Stuck in Diem Contract

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | MEDIUM | IVXDiem.sol: 254,255,531,532,533 | Acknowledged |

## Description

The function `convertFrom18AndRoundDown`

```
convertFrom18AndRoundDown
```

## Recommendation

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

## Resolution

Pending resolution.

# M-23 | Malicious User Can Push Deposits

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| DoS | MEDIUM | IVXQueue.sol: 161 | Acknowledged |

## Description

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

## Recommendation

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

## Resolution

Pending resolution.

# M-24 | Impossible to Close Option

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| DoS | MEDIUM | IVXOracle.sol: 191 | Acknowledged |

## 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.

```
settleOptionsExpired
getSpotPriceAtTime
```

## Recommendation

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

## Resolution

Pending resolution.

# L-0 | No Price Limit on Swaps

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | LOW | IVXExchange.sol: 98 | Acknowledged |

## 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.

## 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

Pending resolution.

# L-1 | ID of 0 is valid for the idModule modifier

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Validation | LOW | IVXDiemToken.sol: 54 | Acknowledged |

## 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.

```
idModule
```

## Recommendation

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

```
idModule
```

## Resolution

Pending resolution.

# L-2 | averageEntry is not being updated on full close trade

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | LOW | IVXPortfolio.sol: L238 | Acknowledged |

## 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

## Resolution

Pending resolution.

# L-3 | Unnecessary block.timestamp Emitted

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Optimization | LOW | IVXQueue.sol: 178, 309 | Acknowledged |

## 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.

```
block.timestamp
```

## Recommendation

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

```
DepositQueued
WithdrawQueued
```

## Resolution

Pending resolution.

# L-4 | Misnamed Variable

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Typo | LOW | IVXOracle.sol: 72 | Acknowledged |

## Description

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

```
setValues
EncodedData
decodedData
```

## Recommendation

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

```
EncodedData
decodedData
```

## Resolution

Pending resolution.

# L-5 | EnumerableSet Should Be Used

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Improvement | LOW | IVXDiemToken.sol: 141-151 | Acknowledged |

## 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.

```
IVXDiemToken
underlyings
```

## Recommendation

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

```
EnumerableSet
underlyings
```

## Resolution

Pending resolution.

# L-6 | Superfluous Processed Check

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Superfluous Code | LOW | IVXQueue.sol: 131-134 | Acknowledged |

## 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.

```
depositsProcessed
withdrawalsProcessed
processCurrentQueue
_rolloverEpoch
currentEpochId
epochData
processCurrentQueue
```

## Recommendation

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

```
depositsProcessed
withdrawalsProcessed
```

## Resolution

Pending resolution.

# L-7 | Inefficient Use Of Storage Variable

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Optimization | LOW | IVXQueue.sol: 128 | Acknowledged |

## Description

A stack variable is stored for the `currentEpochId` storage variable, however the `currentEpochId` in

```
currentEpochId
currentEpochId
```

## Recommendation

Use the `_currentEpochId` stack variable on line 128.

```
_currentEpochId
```

## Resolution

Pending resolution.

# L-8 | Missing Events For reduceQueued Functions

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Events | LOW | IVXQueue.sol: 181, 312 | Acknowledged |

## Description

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

```
reduceQueuedWithdrawal
reduceQueuedDeposit
```

## Recommendation

Implement events for the reduceQueuedWithdrawal and reduceQueuedDeposit functions.

## Resolution

Pending resolution.

# L-9 | Typo

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Typo | LOW | IVXDiem.sol: 246 | Acknowledged |

## Description

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

```
was already transfered to this
          contract
transfered
```

## Recommendation

Correct the spelling error.

## Resolution

Pending resolution.

# L-10 | Blacklist Warning

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Blacklist | LOW | IVXDiem.sol: 526 | Acknowledged |

## 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.

```
treasury
staker
lp
```

## 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

Pending resolution.

# L-11 | assetMarginParams Set For An Unsupported Asset

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Validation | LOW | IVXRiskEngine.sol: 94 | Acknowledged |

## 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.

```
changeAssetMarginParams
_asset
```

## Recommendation

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

```
supportedAssets[_asset]
true
```

## Resolution

Pending resolution.

# L-12 | Parity Check Optimization

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Optimization | LOW | IVXDiemToken.sol: 132 | Acknowledged |

## Description

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

```
createOption
i % 2 == 0
i & 1 == 0
```

## Recommendation

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

```
i & 1 == 0
```

## Resolution

Pending resolution.

# L-13 | Too Many Options Is Not A Good Thing

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Warning | LOW | Global | Acknowledged |

## 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.

## 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

Pending resolution.

# L-14 | Inaccurate Variable Names

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Naming | LOW | IVXRiskEngine.sol: 277-287 | Acknowledged |

## Description

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

```
Vega_shockLoss
Delta_shockLoss
```

## Recommendation

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

## Resolution

Pending resolution.

# L-15 | Liquidity Amount Does Not Include Fee

| CATEGORY | SEVERITY | LOCATION | STATUS |
| --- | --- | --- | --- |
| Logical Error | LOW | IVXQueue.sol: 374 | Acknowledged |

## Description

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

```
LP.withdrawLiquidity
```

## Recommendation

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

```
_amount
_amount-_fee
```

## Resolution

Pending resolution.

# Summary of Recommendations

Based on our comprehensive audit, we provide the following prioritized recommendations to improve the security posture of IVX.

## Priority Matrix

| ISSUE ID | TITLE | SEVERITY | PRIORITY |
|---|---|---|---|
| C-0 | Required Margin Miscalculated | CRITICAL | Immediate |
| C-1 | Liquidations Halted Due To DoS | CRITICAL | Immediate |
| C-2 | Sell Premium Paid Twice | CRITICAL | Immediate |
| C-5 | Incorrect Approval Prevents Liquidation | CRITICAL | Immediate |
| C-6 | Liquidation will fail due to insufficient funds | CRITICAL | Immediate |
| H-0 | vegaDifference Fee Not Valued At The Asset Price | HIGH | High |
| H-1 | Borrowing Fees Accounted For Twice | HIGH | High |
| H-2 | hedgerTotalLiq Errantly Counted As Debt | HIGH | High |
| H-3 | Users Can Withdraw Reserved Utilized Collateral | HIGH | High |
| H-4 | Users Can Avoid Borrowing Fees | HIGH | High |

## General Security Best Practices

- ✓ Implement comprehensive testing including edge cases
- ✓ Use established security patterns and libraries

# Audit Team

## Team Credentials

Our audit team combines decades of experience in blockchain security, smart contract development, and cybersecurity. Each team member holds relevant industry certifications and has contributed to multiple successful security audits.

## Methodology & Standards

Our audit methodology follows industry best practices and standards:

- ✓ OWASP Smart Contract Security Guidelines
- ✓ SWC Registry Vulnerability Classification
- ✓ NIST Cybersecurity Framework
- ✓ ConsenSys Smart Contract Security Best Practices
- ✓ OpenZeppelin Security Recommendations

## Audit Process

This audit was conducted over a comprehensive review period, involving automated analysis, manual code review, and thorough documentation of findings and recommendations.

# Disclaimer & Legal Notice

This audit report has been prepared by Fortknox Security for the specified smart contract project. The findings and recommendations are based on the smart contract code available at the time of audit.

## Scope Limitations

- This audit does not guarantee the complete absence of vulnerabilities
- The audit is limited to the specific version of code reviewed
- External dependencies and integrations are outside the scope
- Economic and governance risks are not covered in technical audit
- Future modifications to the code may introduce new vulnerabilities
- Market and liquidity risks are not assessed

## Liability Statement

Fortknox Security provides this audit report for informational purposes only. We do not provide any warranties, express or implied, regarding:

- The absolute security of the smart contract
- The economic viability of the project
- The legal compliance in any jurisdiction
- Future performance or behavior of the contract
- Third-party integrations or dependencies

# Legal Terms & Usage Rights

## Usage Rights

This audit report may be used by the client for:

- ✓ Public disclosure and transparency
- ✓ Marketing and promotional materials
- ✓ Investor due diligence processes
- ✓ Regulatory compliance documentation
- ✓ Technical documentation and reference
- ✓ Security assessment presentations
- ✓ Community transparency initiatives

## Restrictions

The following restrictions apply to this report:

- ✓ Report content may not be modified or altered
- ✓ Fortknox Security branding must remain intact
- ✓ Partial excerpts must maintain context and accuracy
- ✓ Commercial redistribution requires written permission
- ✓ Translation must preserve technical accuracy

## Intellectual Property

This report contains proprietary methodologies and analysis techniques developed by Fortknox Security. The format, structure, and analytical approach are protected intellectual property.

## Contact Information

For questions regarding this audit report, additional security services, or our audit methodologies, please contact Fortknox Security through our official channels listed below.

**Fortknox Security**

🌐 https://www.fortknox-security.xyz

🐦 @FortKnox_sec

📧 support@fortknox-security.xyz


# FORTKNOX SECURITY

Web3 Security at Fort Knox Level


## Contact Us

🐦 @FortKnox_sec

📱 @FortKnox_sec

🌐 fortknox-security.xyz

📧 support@fortknox-security.xyz


Audit performed by
Fortknox Security