Xave

Introduction

This report provides a comprehensive analysis of Xave Finance's smart contract architecture and risks. Its goal is to evaluate the protocol's design, identify vulnerabilities, and suggest mitigations to enhance security and resilience.

Protocol Description

Xave Finance is a decentralized finance (DeFi) platform that bridges traditional finance concepts with blockchain technology. It focuses on facilitating efficient and scalable markets for tokenized assets, such as **stablecoins** and **Real-World Assets (RWA)** like commodities or fiat currencies. Its innovative architecture and products aim to solve liquidity and efficiency challenges in decentralized exchanges (DEXs).

Protocol Architecture

1. FXPools are liquidity pools specifically designed to enable efficient swaps between tokenized currencies, such as stablecoins or other tokenized real-world assets.
    a. How It Works:
        - Uses a unique FXInvariant bonding curve optimized to mimic traditional FX markets.
        - Maintains liquidity health by adjusting pool ratios dynamically based on market activity.
        - Relies on Chainlink price oracles to provide accurate, real-world price data for swaps.
    b. Goal:
        - Increases capital efficiency, making it 20x more effective than traditional stableswap mechanisms.
        - Supports the adoption of non-USD stablecoins, diversifying currency markets on the blockchain.

2. Lending Market is a decentralized lending platform integrated into Xave, allowing users to borrow stablecoins pegged to local currencies.

   a. How It Works:

   - Users deposit assets like ETH, WBTC, or stablecoins as collateral.
   - Borrowing is over-collateralized, ensuring loans are secured.
   - Interest rates are dynamically set using an adaptation of Aave V2's utilization curve.
   - Utilizes Chainlink oracles for accurate valuation of collateral and borrowed assets.

   b. Goal:

   - Fills a unique niche in DeFi by focusing on regional currencies, which are often overlooked.
   - Enables more inclusive access to decentralized borrowing across global markets.

3. Governance with XAV Token: The XAV token is the native governance token of the Xave ecosystem, similar to Balancer's BAL token.

   a. How It Works:

   - Holders can lock XAV to mint veXAV (vote-escrow XAV), which grants governance rights.
   - veXAV holders:
     - Vote on FXPool parameters (e.g., liquidity fees, pool incentives).
     - Influence the distribution of XAV token emissions to liquidity providers.
     - Earn a portion of the protocol's fees generated from trading activity.

   - Goal:
     - Empowers the community to shape the protocol's future.
     - Aligns incentives between liquidity providers, stablecoin issuers, and token holders.
     - Encourages long-term commitment by rewarding users who lock their tokens.

4. Balancer Vault Integration: The Balancer Vault serves as a critical component in Xave's architecture, underpinning the operation of FXPools and securing liquidity management.

   a. How It Works:

   - Acts as the liquidity backend for FXPools, consolidating asset management.
   - Manages approximately $4 million in tokenized assets to ensure sufficient liquidity.
   - Provides optimized routing for token swaps, reducing gas costs and increasing operational efficiency.
   - Leverages Balancer's extensively audited infrastructure to ensure security and reliability.

   b. Goal:

   - Enhances scalability by providing a robust and flexible framework for expanding liquidity pools.
   - Reduces operational overhead and ensures efficient fund utilization.
   - Strengthens protocol security by relying on Balancer's proven vault model.

Audit Reports

Xave Finance ensures the security of its smart contracts through a multi-phase process. Contracts are first deployed on public and private testnets for thorough integration and unit testing. After passing these tests, they move to a preproduction mainnet instance to validate performance under near-live conditions. Before production deployment, Xave conducts bug bounty programs to identify potential vulnerabilities and engages third-party security auditors, to review the code. Only after addressing all identified concerns are the contracts deployed to the live environment.

Price Oracles Audit

Audit conducted by Certik.

Audit link.

Key Findings:

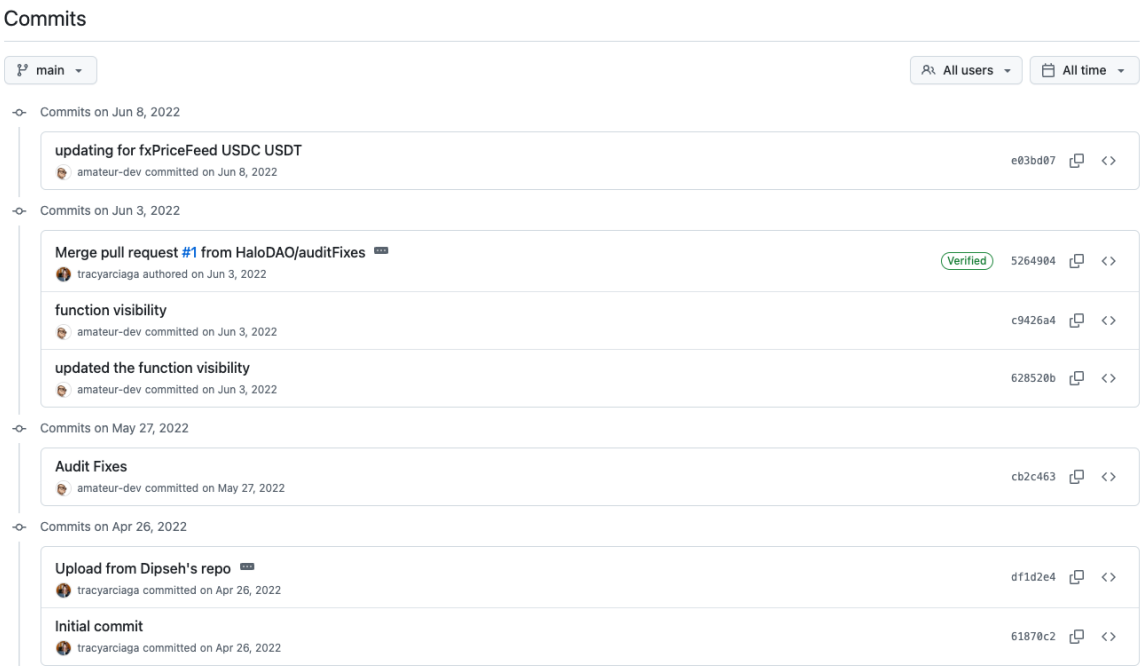
1. Resolved Issues:

- SafeMath Usage: Initially, arithmetic operations lacked safeguards against underflow and overflow attacks. This was resolved by upgrading the Solidity version to 0.8.4, which includes built-in protections.
- SafeCast Implementation: Typecasting operations were updated to use the SafeCast library, preventing overflow errors when converting `uint256` to `int256`.

2. Informational Observations:
   - Use of Decentralized Oracle: Chainlink price feeds were acknowledged as reliable. It was suggested to document the data sources clearly to inform users of associated risks.
   - Gas Optimization: Recommendations included marking certain functions as `external` and declaring variables as `immutable` to reduce gas costs.
3. Scope and Review Process:
   - Two contracts were assessed: `HLPPriceFeedOracle` and `PriceFeed`.
   - Issues found during the review were categorized into medium (e.g., SafeCast usage) and informational (e.g., naming conventions), with no critical or major vulnerabilities identified.

Price Oracle code audit review and fixes

Price Oracle code audited by Certik



code audit review and fixes

Even though the audit findings were mitigated, the Price Oracle repository has not been updated in the last two years. Additionally, the Price Oracle addresses deployed on the Avalanche C-Chain do not match the code audited by Certik, creating a gap in context regarding what has transpired over the past two years.

***@TODO:** Task: Ask the Xave project team about the Price Oracle code currently deployed on the Avalanche C-Chain and provide references to the audit.*

Chainlink Price Oracle Address on Avalanche (C-Chain)

Contract	Address
USDC/USD Price Feed	https://snowtrace.io/address/0xF096872672F44d6EBA71458D74fe67F9a77a23B9
EUR/USD Price Feed	https://snowtrace.io/address/0x192f2DBA961Bb0277520C082d6bfa87D5961333E
CHF/USD Price Feed	https://snowtrace.io/address/0xA418573AB5226711c8564Eeb449c3618ABFaf677

FXPool Audit

Audit conducted by Akira Tech and PeckShield.

Audit link.

Key Findings:

1. Resolved Issues

- SafeMath Usage (Akira Tech):
  - Arithmetic operations in some contracts were initially exposed to underflow and overflow risks. Akira Tech highlighted the lack of explicit safeguards, recommending updates to Solidity 0.8.4, which includes built-in protections. This was implemented successfully.
- SafeCast Implementation (PeckShield):
  - PeckShield identified issues with typecasting operations, specifically converting `uint256` to `int256`. These were updated to use the SafeCast library, mitigating potential overflow risks.

2. Informational Observations

- Use of Decentralized Oracle (Akira Tech):
  - Akira Tech recognized Chainlink price feeds as a robust and decentralized solution. However, they emphasized the importance of documenting the sources and potential risks of relying on external oracle data, such as stale or invalid data due to delays.
- Gas Optimization (PeckShield):
  - PeckShield recommended minor optimizations to reduce gas costs, including marking certain functions as `external` and declaring variables as `immutable`. These adjustments enhance efficiency but do not impact security.

3. Scope and Review Process

- Akira Tech Audit:
  - Scope: Focused on Xave Finance AMM contracts, including `FXPool`, `Assimilators`, and supporting libraries like `CurveMath` and `ABDKMath64x64`.
  - Key Identified Issues:
    - Oracle price validation using `Chainlink.latestAnswer` lacked proper checks for staleness, which could lead to outdated pricing.

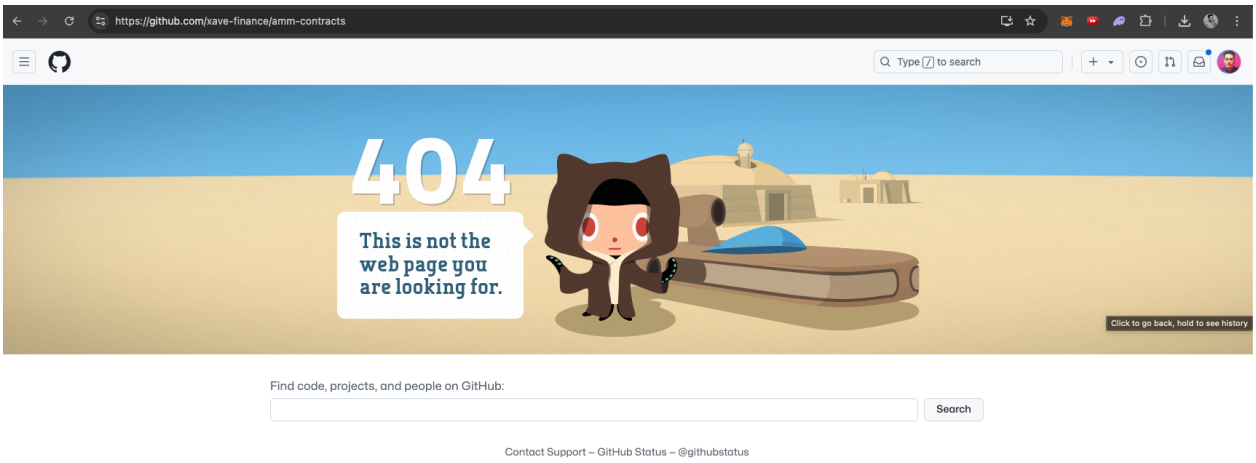
        - Redundant `require` checks in `FXPool.onSwap` and insufficient initialization safeguards.
    - Review Outcome: Issues categorized as minor and major, all resolved.
- PeckShield Audit:
    - Scope: Covered the FXPool-based stablecoin AMM built on Balancer V2 Vault, focusing on contracts like `BaseToUsdAssimilator` and `FXPool`.
    - Key Identified Issues:
        - Critical caller validation gaps in `onJoinPool` / `onExitPool` were fixed to enhance access control.
        - Business logic flaws in `outputRaw` of `BaseToUsdAssimilator` and `UsdcToUsdAssimilator` were corrected to ensure accurate value transfers.
    - Review Outcome: 5 findings categorized as critical, medium, low, and informational, all resolved or mitigated.

Even though the findings from Akira Tech and PeckShield were mitigated, as stated in their respective reports, the repository they audited no longer exists (https://github.com/xave-finance/amm-contracts).

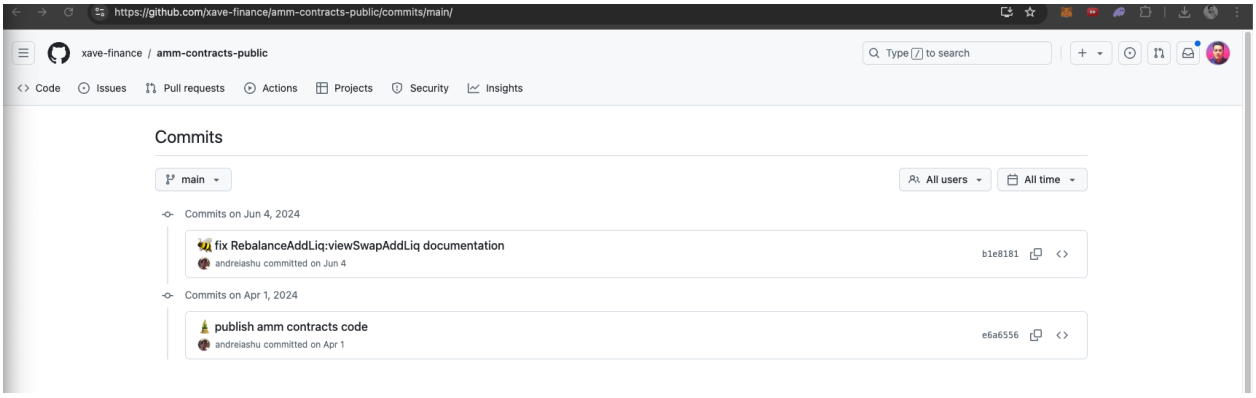
A similar repository was created on April 1st (https://github.com/xave-finance/amm-contracts-public/commits/main/), with its first commit appearing on that date. However, it is unclear if this new repository contains the same secure code that was audited. Both audit reports reference specific commit hashes, which were intended to ensure the code's integrity and security as reviewed at that point in time.

Without verifying the commit history or matching the code in the new repository to the audited commit hashes, it is imprecise and potentially misleading to claim that the code is secure based solely on the previous audit reports.

***@TODO:** Confirm with the Xave project team whether the newly created repository (first commit on April 1st) aligns with the commit hashes referenced in the Akira Tech and PeckShield audit reports. If not, request clarification on the differences and provide references to the audited code.*



Code audited by Akira Tech and PeckShield



Similar code

Lending Market

Audit conducted by Akira Tech.

Audit link.

Repo.

Key Findings:

1. Resolved Issues:
    - Price Manipulation Vulnerability in `buybackRnbw()`:The `buybackRnbw()` function was initially susceptible to price manipulation due to the absence of a minimum output token enforcement. This was resolved by introducing a minimum token output

parameter (`minRNBWAmount`) and leveraging the `Uniswap V2` `getAmountsOut` function to calculate fair prices prior to swaps.

- Gas Optimization in `WETH9` Variable:The `WETH9` address was updated to a constant variable, optimizing gas costs during contract interactions.

2. Informational Observations:
   - Usage of OpenZeppelin Libraries:Some libraries like `SafeMath` and `Context` were redundantly reimplemented. The recommendation to reuse official OpenZeppelin implementations has been adopted to enhance maintainability.
   - Unnecessary Deadlines in Swap Functions:The use of `block.timestamp + 60` in deadline parameters was deemed excessive. Simplifying this to `block.timestamp` was recommended for better clarity without gas penalties.
3. Scope and Review Process:
   - Reviewed Components:The primary focus was on the `Treasury` contract and related interfaces (`ICurve`, `IUniswapV2Router01/02`) within the Xave Lending Market repository.
   - Review Methods:The audit covered manual code reviews targeting security risks such as re-entrancy, gas optimization, and architectural clarity. A total of 15 person-days were invested over two weeks.
   - Severity Distribution:
     - Informational: 1 (closed)
     - Minor: 3 (closed)
     - Medium: 1 (closed)
     - Major: 0
4. Recommendations and Future Improvements:
   - Increase test coverage to approach 100%, ensuring both positive and negative test scenarios.
   - Remove obsolete modifiers (e.g., `onlyEOA` in the `Treasury` contract) to streamline the codebase.

The commit hash referenced in the audit report dates back to August 17, 2021 (view commit). However, the code repository has received additional commits since that date.

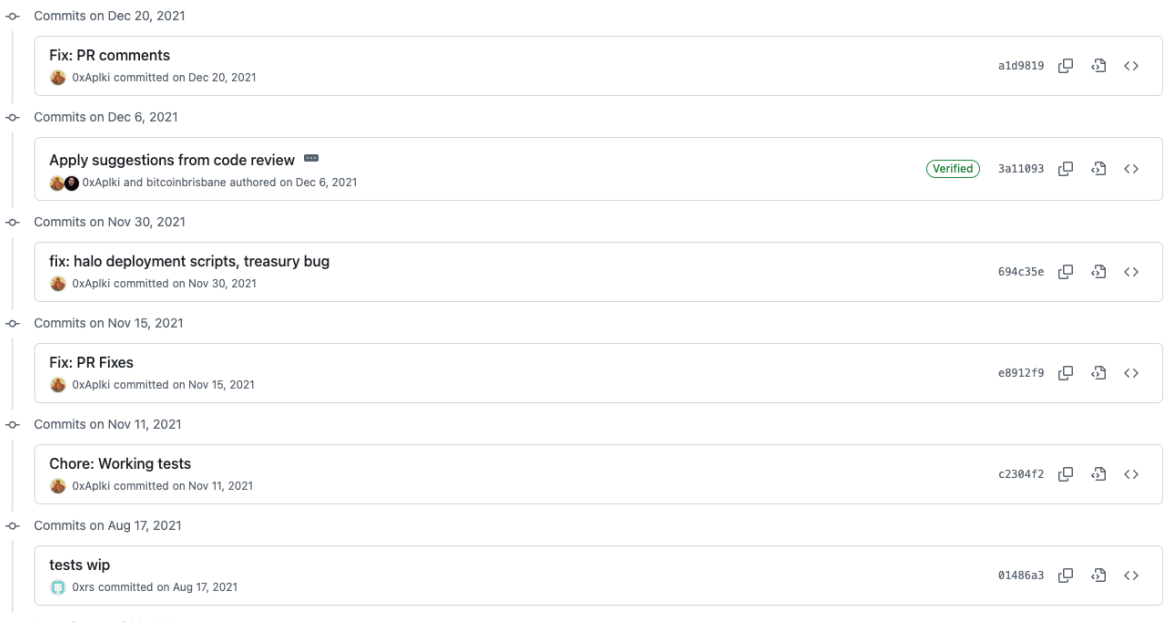
For instance, the `contracts/buyback/Treasury.sol` *contract has undergone further updates between August 17, 2021, and December 2021, as evidenced by subsequent commits.*

It is unclear whether all these commits address the issues and recommendations highlighted in the audit conducted by Akira Tech. Without explicit confirmation or documentation linking these changes to the audit report, it cannot be definitively stated that the code is fully protected against potential vulnerabilities.

Given that the audit referenced main protocol contracts, additional verification of these later commits is necessary to ensure all findings and recommendations from the audit have been properly implemented and that the protocol remains secure.

@TODO:

1. Confirm with the Xave project team whether the additional commits made between **August 17, 2021**, and **December 2021** in the `contracts/buyback/Treasury.sol` file were directly related to resolving issues raised in the Akira Tech audit report.
2. Request a detailed changelog or documentation linking these commits to the specific audit findings for verification.
3. Validate if any unrelated changes during this period could introduce new vulnerabilities.
4. Ensure all updates are reviewed and tested for compliance with the audit's recommendations and general security best practices.



Treasury.sol contract commits

Identification of Potential Risks

1. Technical Risks:
    a. Reentrancy Attacks: Risks in FXPool swaps due to improper state handling.
        - Example: Mention DAO Hack and steps to mitigate with ReentrancyGuard or Checks-Effects-Interactions pattern.
    b. Improper Access Control: Risks in governance mechanisms like veXAV voting.
        - Mitigation: Use OpenZeppelin's AccessControl library for robust role definitions.
2. Economic Risks:
    a. Liquidity Imbalances: Risks from arbitrage activities destabilizing FXPools.
    b. Collateral Risks: Potential losses from undercollateralized loans.
3. Operational Risks:
    a. Oracle Dependency: Centralization risks tied to Chainlink outages or price delays.
        - Mitigation: Propose multi-oracle setups.
    b. Upgradeability Risks: Risks of introducing vulnerabilities during contract updates.

- Example: Compromise in upgrade keys or improper testing.

4. Vault Risks:
    a. Concentration Risk: The Balancer Vault manages approximately $4 million in tokenized assets. A compromise of the Balancer infrastructure could severely impact Xave's operations.
        - Mitigation: Establish fallback liquidity mechanisms and consider diversifying liquidity across multiple vaults or protocols.
    b. Dependency on Balancer Upgrades: Xave's reliance on Balancer's infrastructure means that any vulnerabilities, delays, or failures in Balancer could directly affect its functionality.
        - Mitigation: Monitor Balancer Vault updates closely and develop rapid response plans for potential issues.
5. User Risks:
    a. Complexity in FXPool Mechanics: Misunderstanding the FXInvariant model could lead to financial losses.
    b. UI/UX Challenges: Issues with wallet connections or unclear instructions.

Risk Level Grading

Risk Type	Description	Level	Justification
Reentrancy Attacks	FXPool swaps may lack sufficient checks	High	Direct financial loss potential
Oracle Downtime	Dependency on single oracle system	Medium	Limited backup mechanisms
Liquidity Imbalance	Arbitrage draining pools during volatility	High	Historical examples in similar protocols
Governance Centralization	Large holders control decisions	Medium	Potential for collusion
User Misuse	Complex mechanics in FXPools	Low	Limited impact on overall protocol health

Recommendations

1. Technical Improvements:

- Enforce stricter testing protocols using fuzzing tools like Echidna for FXPool contracts.
- Enhance audit schedules, especially post-upgrade, and ensure multiple auditors are engaged.
- Develop fallback mechanisms for Chainlink oracles using decentralized alternatives like Tellor.

2. Economic Safeguards:
   - Adjust FXPool liquidity thresholds to minimize imbalances.
   - Implement penalty fees for trades moving liquidity out of the "beta region."
3. Operational Enhancements:
   - Increase transparency in veXAV governance by publishing regular token distribution reports.
   - Regularly review the allocation of funds in the Balancer Vault to ensure it aligns with protocol usage and risk appetite.
   - Establish liquidity mirroring in vaults.
4. Education and Communication:
   - Publish step-by-step guides for FXPool use cases.
   - Simulate FXInvariant trades in a sandbox environment for user education.

Conclusion and Ratings

This report has evaluated the architecture, security practices, and audit references of Xave Finance's smart contracts. Below is a rating for each aspect on a scale of 1 to 5, along with final observations and recommendations.

1. Protocol Architecture: 4/5

Xave Finance demonstrates a robust and innovative protocol architecture, with clear modular components such as FXPools, the Lending Market, and Governance via the XAV token. The integration of the Balancer Vault and Chainlink oracles enhances efficiency and security. However, the reliance on specific external infrastructures introduces risks that require ongoing monitoring and mitigation strategies.

2. Audit Coverage: 2/5

While historical audits were conducted by reputable firms such as Certik and Akira Tech, the reports appear outdated for several components. The Price Oracle repository has not been updated in over two years, and discrepancies exist between the audited code and the currently deployed contracts on Avalanche C-Chain. Additionally, the new Lending Market V1 repository (GitHub link) lacks references to any audits in the Contract Audits tab, leaving critical gaps in transparency and security assurance.

3. Security Practices: 2/5

Xave employs commendable security practices, such as testnet deployments, preproduction validations, and bug bounty programs. However, the absence of updated audits and the lack of documentation for recent changes in repositories create vulnerabilities in the overall security framework. The dependency on external oracles and Balancer Vault also increases potential risks.

4. Risk Identification and Mitigation: 2/5

The protocol's potential risks have been identified and categorized effectively, including reentrancy, liquidity imbalances, and oracle dependency. The proposed mitigations, such as fallback mechanisms and liquidity mirroring, provide viable solutions. However, implementing these recommendations consistently across the protocol remains critical.

5. User Transparency and Documentation: 2.5/5

While Xave provides technical documentation and references for older audits, the lack of alignment between the latest repositories and documented audits reduces transparency. Users and developers need more accessible and up-to-date references to ensure trust and clarity.