SUBLIME

Security Assessment

Hourglass v2

Audit Summary Report

Jul 20, 2023

Table of contents

Document revisions. 3

Introduction to Sublime Group: 4

- Technical Security Assessment and Advisory Services: 4
- Smart Contract Auditing: 4
- Decentralized Finance (DeFi): 5
- Quantitative Trading: 5
- Market Making: 5

Audit Test and Reporting Disclaimer: 6

Audit summary 7

Findings Summary 11

List of Issues Found: 13

- ISSUE-1 Not working before/after transfer hook 13
- ISSUE-2 Libraries not in a stable release version 14
- ISSUE-3 Typo in TwoStepOwnableInterface 15
- ISSUE-4 Leftover code in GeneralMatureHoldingVault 16
- ISSUE-5 MigrateToMaturedVault reentrancy 17

Test coverage: 18

Static analysis 19

Integration with Other Protocols 20

- Enhancing Interoperability: 20
- Risk Assessment: 20
- Ongoing Risk Management: 20

Document revisions.

Revision	Description	Date
1.0	Initial report	Jul 20, 2023

Introduction to Sublime Group:

Sublime Group is a leading organization revolutionizing the financial landscape through its expertise in Decentralized Finance (DeFi), quantitative trading, market making, and technical security assessment. With a commitment to innovation and trust, Sublime Group offers cutting-edge solutions and advisory services in the rapidly evolving digital asset ecosystem.

Technical Security Assessment and Advisory Services:

Sublime Group excels in providing cutting-edge technical security assessment and advisory services. Leveraging advanced tools and methodologies, we conduct comprehensive security assessments, penetration testing, and vulnerability analysis to fortify systems and infrastructure. Our team of highly skilled cybersecurity experts possesses extensive expertise in identifying and mitigating potential risks and ensuring compliance with regulatory frameworks. By partnering with Sublime Group, clients benefit from our industry-leading security solutions, enabling them to safeguard their digital assets and maintain a robust security posture that surpasses competitors.

Smart Contract Auditing:

At Sublime Group, we pride ourselves on delivering meticulous smart contract audits that go beyond industry standards. Our experienced auditors combine their deep understanding of blockchain technology with an arsenal of cutting-edge tools and advanced methodologies. In addition to manual code analysis, our auditors utilize static code analysis tools to perform automated checks, machine learning techniques to identify complex vulnerabilities, and comprehensive fuzz testing to detect potential security loopholes. By employing these advanced methods, we thoroughly review the reliability, functionality, and security aspects of smart contract code. Our auditors provide actionable recommendations to address identified vulnerabilities, ensuring the robustness and integrity of our clients' blockchain-based applications. Sublime Group's smart contract auditing process, powered by state-of-the-art LLM and fuzzing tools, puts Sublime ahead of the industry standard. By choosing Sublime Group, clients benefit from our unrivaled expertise and innovative approach, guaranteeing the utmost security for their smart contracts and staying ahead of the curve in the ever-evolving landscape of software security.

Decentralized Finance (DeFi):

Sublime Group leverages decentralized protocols and smart contracts to provide seamless access to decentralized lending, borrowing, yield farming, and decentralized exchanges. Our solutions bridge traditional finance with the blockchain ecosystem, empowering users in the DeFi space.

Quantitative Trading:

Sublime Group's skilled quantitative traders optimize trading strategies using advanced algorithms and data analysis techniques. Our expertise in market dynamics, liquidity, and risk management enables efficient execution and enhanced trading performance across digital asset markets.

Market Making:

Sublime Group ensures market liquidity, reduces spreads, and minimizes price volatility through our market-making services. Our proprietary algorithms and risk management systems contribute to fair and efficient price discovery, benefiting institutional and retail investors.

Audit Test and Reporting Disclaimer:

Sublime Group conducted activities for this project according to the statement of work. However, it is important to note that security assessments have time constraints and rely on client-provided information. Thus, the findings in this report may not encompass all security issues or flaws in the system or codebase.

Sublime Group employs automated testing techniques and manual security reviews to assess software controls. However, automated tools have limitations, such as not capturing all edge cases or incomplete analysis within time limits. These limitations are subject to project time and resource constraints.

Clients should understand that while Sublime Group's test coverage is comprehensive within the project's scope, it may not uncover all potential vulnerabilities or flaws. The audit scope does not cover code provided by third party libraries or protocols that hourglass integrates with. However, we have made due diligence when conducting our test to check for any known vulnerabilities and correct usage of said libraries. Ongoing security assessments and proactive measures are advised to maintain system integrity. Sublime Group remains dedicated to assisting clients in enhancing their security and providing expert guidance throughout the development process.

Audit summary

Hourglass engaged Sublime Group to review the security of its updated v2 smart contracts. Our dedicated team has invested 3 person-days of effort into conducting a security review of the client-provided source code. We have conducted a thorough audit of the smart contracts code provided by Hourglass.

The audit commenced with revision fa5fe4d3333d0cbc1ae3c868fcd2f1547f85513a on the v2 branch, which served as the starting point for our evaluation.

Throughout the audit process, our team has diligently reviewed the Hourglass smart contracts, carefully scrutinizing the codebase for potential security vulnerabilities, logical flaws, and adherence to best practices. Our objective was to assess the robustness and integrity of the contracts, identifying any potential issues that could compromise the security and functionality of the system.

As part of the audit work, we engaged in a collaborative effort with Hourglass, ensuring a smooth and effective communication channel between our team and theirs. This allowed for the timely resolution of any issues or concerns that arose during the audit. Hourglass actively participated in addressing the identified issues, promptly implementing fixes and enhancements based on our recommendations.

The audit process involved a comprehensive analysis of the codebase, including but not limited to the smart contracts and associated libraries or dependencies. Our experienced auditors meticulously reviewed the code, examining key aspects such as contract architecture, data handling, access controls, input validation, and adherence to industry best practices.

During the audit, we uncovered several issues, which were promptly communicated to the Hourglass team. We provided detailed reports outlining the identified vulnerabilities and flaws, along with recommended remediation measures. Hourglass demonstrated a strong commitment to security and swiftly addressed the identified issues by implementing the recommended fixes. This collaborative approach between our team and Hourglass ensured a robust and secure smart contract implementation.

It is important to note that the audit conducted on Hourglass smart contracts was focused solely on the code provided by Hourglass. While we diligently assessed the code for potential vulnerabilities, logical flaws, and adherence to best practices, it is essential to acknowledge that unforeseen risks or vulnerabilities may exist beyond the scope of the audit.

Hourglass's proactive engagement during the audit process, including addressing the identified issues, reflects their dedication to ensuring the security and reliability of their smart contracts. By actively participating in the audit and promptly resolving any identified issues, Hourglass showcases a commitment to delivering a secure and trustworthy platform for their users.

In conclusion, the audit conducted on the Hourglass smart contracts involved a comprehensive review of the codebase, analysis of potential vulnerabilities, and collaboration with the Hourglass team to address and rectify identified issues. The proactive engagement from Hourglass in fixing the identified issues demonstrates their commitment to maintaining a secure and robust smart contract implementation.

The contracts were updated by hourglass multiple times during the audit and re-checked by Sublime Group on later revision numbers. The list of all revisions checked during the audit can be found below:

1. fa5fe4d3333d0cbc1ae3c868fcd2f1547f85513a
2. ee2fb780bcfbc312a5be92026ed16272af145df8
3. c201f8dccb3aeef90f19272b74caf8f26aa9f9af
4. F12d31817307ade84d6443c0a9639a30f103e6ee
5. 9985b97a4ab7e486f84a3a7e747b6ee7769f4461
6. 7df7716d5d1e3182630ae56d6612b277d8d4a37e

Below is a list of smart contracts included in the audit scope along with their respective sha256 checksums of the latest revision tested:

Filename	SHA-256
FeeManager.sol	093509fdaaf881926471c111a7349567992ea050ccb4db8cb93b603c4d638d98

HourglassCustodian.sol	5caea62ea32bbe9cb87dd3bba8db3b6fdab8910b91bfb7d6001f229690090536
HourglassCustodianV2.sol	150ed1a02fe906d8f67a721c0d46a82a0ba376cfaec5ea1ada22c0feb03f9832
RewardsDistributor.sol	d7d8249d0ec9af66a39e3001efe2bcc2b3c4fdcbec904260a0610d99884cb72c
RewardsDistributorV2.sol	86389a8e8b1ea60bed91650169d3b62f029a6c302b8a87acf55e3b8b1757c55f
ConvexFraxVault.sol	dd0dd7588ccb5918968995810bc5a838a0dbda7f41b8309c218bef3ad7365844
HourglassConvexFraxReceipt.sol	825f65c8b931876b60c4f89011a6b8608650f5cfd4baefffbc41dc7456be1de4
HourglassConvexFraxReceiptV2.sol	d5f0a47b0aec24fcdf9e289cdbb71dc895d1a5b00a70c80a167b23e1f0f388c5
ERC20Intermediary.sol	cfca8d4843a64927fce51cd974bde28d1a8a2fae0b76919f31c894b8fd336134
GeneralMatureHoldingVault.sol	2968af1c4545d0929f30364e671bb0f07bf0bf6ba00f5a9bf7340ad7415a61b6
GeneralMatureHoldingVaultV2.sol	b714e76da6165263f1fb662a88af08ec4b7e8391f51642269cc8a1fc1768a42c
TwoStepOwnable.sol	592a1a798946ec89c01c38949ed419aa13e4c914da6f7810d9dc90a9916097d1
Burn.sol	4a927c2fcbbe085f3141be0c2fe10220e9d0ff3228e51a7d78fa8d08b00376db
HourglassToken.sol	d72d253cc61499a7f43d22bf908e17c47aa8599db1465376cafc48158fcc5a8c

HourglassCustodianTest.t.sol (later renamed to HourglassProtocolTest.t.sol)	a4f55ae5b20c71576aed7ae1644dc7cd80c7029e493e1463cf380358da14f5f2

ConvexFraxMaturedHoldings.sol was not verified due to work in progress on the side of Hourglass.

Findings Summary

Our primary focus during the audit was to identify potential vulnerabilities in the Hourglass smart contracts, specifically related to business logic, arithmetic operations, and integration with other protocols and tokens. Our objective was to ensure that the protocol remains secure, preventing any unauthorized access or fund theft. Additionally, we have provided Hourglass with guidance on clean code practices and industry best practices wherever applicable.

This is a followup to the previous v1 audit and thus certain contracts files were not updated since. Please refer to the previous audit for a comprehensive summary of areas not covered by this document.

Throughout the audit process, we diligently analyzed the smart contracts codebase, resulting in the identification of several issues with varying severity levels. We provided Hourglass with comprehensive guidance and recommendations on security enhancements, best practices, and maintaining clean code standards. Collaboratively, we worked with Hourglass to address and rectify all the issues discovered during the audit.

While evaluating the overall security of the system, we have identified sevral issues, with severity levels ranging from critical to low. We have advised Hourglass to enhance their clean code practices for better maintainability and security.

The test coverage of the protocol was determined to be satisfactory; however, we recommend improving the quality of the tests. A detailed section in the report outlines all the issues identified within the tests.

In conclusion, the audit process has uncovered several issues of varying severity within the Hourglass smart contracts codebase. However, we are pleased to report that the overall security of the system is satisfactory, with no critical issues threatening user funds. We have collaborated closely with Hourglass to address and resolve the identified issues, reinforcing their commitment to maintaining a secure protocol.

Furthermore, we recommend Hourglass to enhance their clean code practices and improve the quality of tests.. By implementing these recommendations, Hourglass can further enhance the security, reliability, and usability of their protocol.

Below is a summary of all the issues found during the audit divided into respective severity categories ranging from critical to suggestion.

Severity	Number of issues	Remaining after Audit
● critical	1	0
● high	0	0
● medium	1	0
● low	2	1
● suggestion	1	1

List of Issues Found:

ISSUE-1 | Not working before/after transfer hook

Severity	Revision found	Status
● Critical	fa5fe4d3333d0cbc1ae3c868fcd2f1547f85513a	resolved by hourglass

Description:

Due to an update of an OpenZeppelin dependency the HourglassDepositReceiptV2 before and after transfer hooks (responsible for Total Supply tracking and automated rewards claiming and processing) stopped working. This is due to a breaking change in how this functionality is handled in the new OZ version. The internal function is missing the override keyword and is not called anywhere in the contract

Recommendation:

We strongly recommend reverting to the last stable release version of the OpenZeppelin smart contracts library as this change is not part of an official release of the OZ library.

```
function _beforeTokenTransfer(
   address,
   address from,
   address to,
   uint256[] memory ids,
   uint256[] memory amounts,
   bytes memory
) internal {

(...)

}
```

ISSUE-2 | Libraries not in a stable release version

Severity	Revision found	Status
● Medium	fa5fe4d3333d0cbc1ae3c868fcd2f1547f85513a	resolved by hourglass

Description:

Some of the libraries used throughout the project are in a most recent development version and not a stable official release version (for example openzeppelin-contracts and openzeppelin-contracts-upgradeable library). This can be dangerous as the changes in the master branch of said dependencies can contain errors or lack proper security audit compared to the official release.

Recommendation:

We strongly recommend using the latest stable release version of the OpenZeppelin smart contracts library (4.9.2 at the time of writing). As a general rule of thumb we recommend using only the official releases of libraries and dependencies and adhering to any recommendations made by the library development teams in that regard.

ISSUE-3 | Typo in TwoStepOwnableInterface

Severity	Revision found	Status
● Low	fa5fe4d3333d0cbc1ae3c868fcd2f1547f85513a	Acknowledged by Hourglass

Description:

TwoStepOwnable has an import statement of TwoStepOwnableInterface. The file however is named TwoStepOwnableinterface, there is a difference where one is using lower and the other upper case letter.

Recommendation:

We recommend fixing this minor issue and making the filename consistent with the import statement.

ISSUE-4 | Leftover code in GeneralMatureHoldingVault

Severity	Revision found	Status
● Low	fa5fe4d3333d0cbc1ae3c868fcd2f1547f85513a	Fixed by Hourglass

Description:

GeneralMatureHoldingVault has some leftover debug console log code. This is not a major issue as it is replaced by GeneralMatureHoldingVaultV2.

Recommendation:

We recommend removing any leftover debug code in order to adhere to clean code standards.

ISSUE-5 | MigrateToMaturedVault reentrancy

Severity	Revision found	Status
● Low	fa5fe4d3333d0cbc1ae3c868fcd2f1547f85513a	Acknowledged by Hourglass

Description:

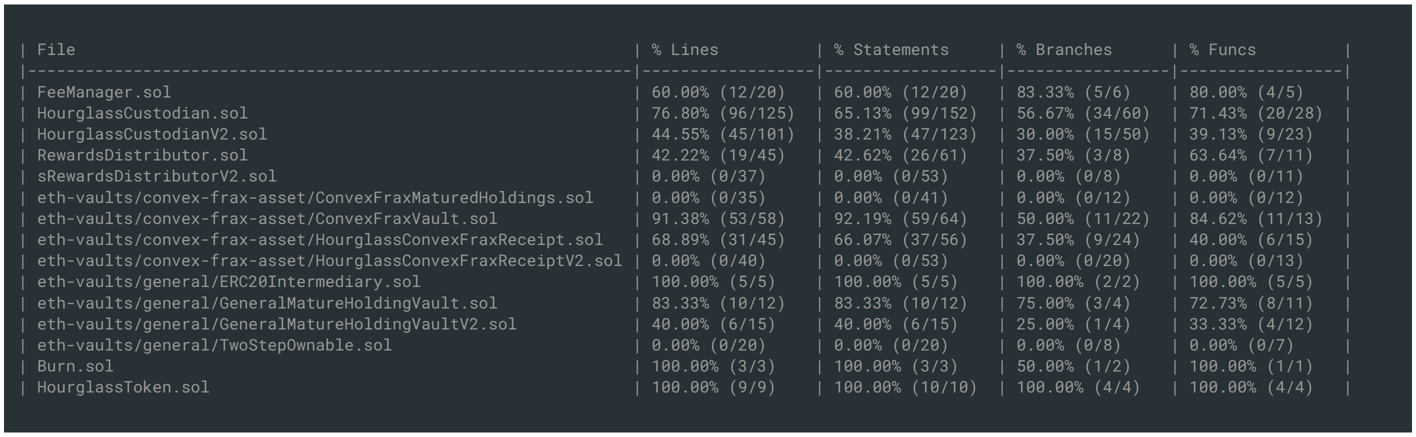
Hourglass custodian function MigrateToMaturedVault is no longer protected against reentrancy. Due to the current state of the implementation and interactions order inside of the contract functions, we have not found any reentrancy or cross function reeentrancy attack vector which is commendable. Nevertheless we feel that it is a good precaution to either make it nonreentrant except for privileged addresses or simply add access control to the function and make it admin only. This is due to the fact that further changes to the protocol might expose this core function to cross function reentrancy.

Recommendation:

We recommend adding some form of reentrancy mitigation to this function.

Test coverage:

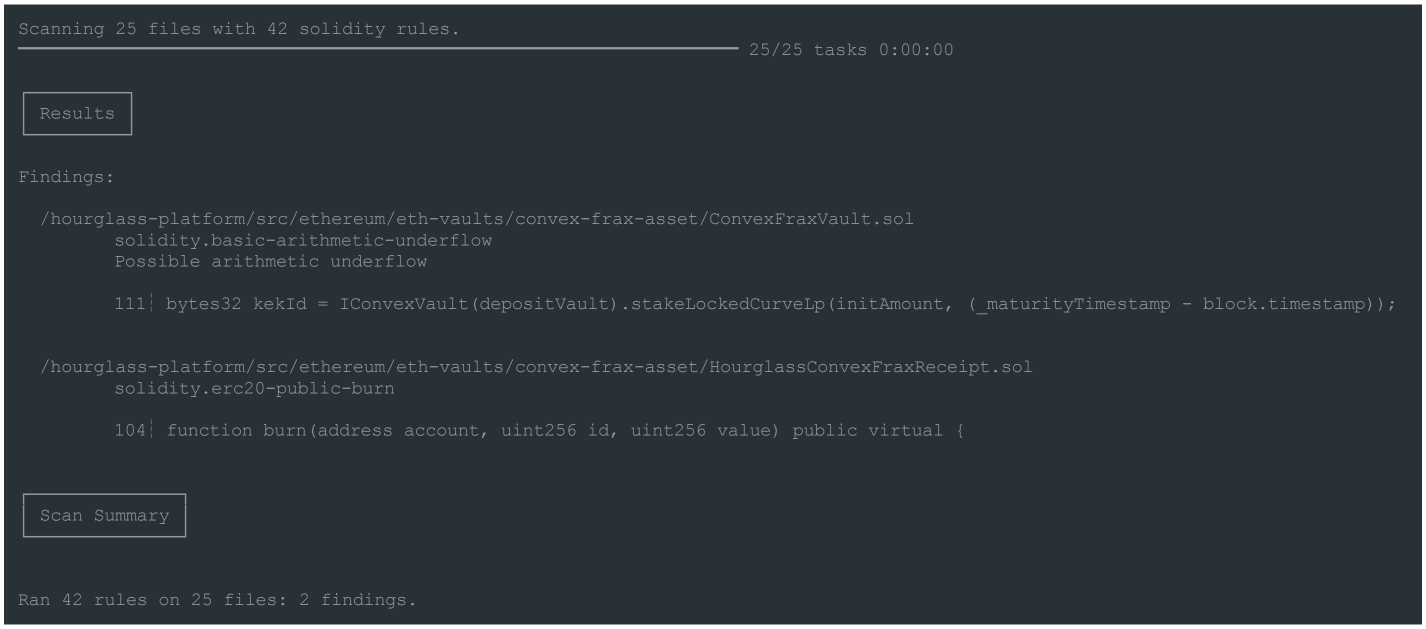
The protocol codebase contained tests located in HourglassProtocolTest.t.sol, UpgradeCustodianV2Test, Hourglass.t.sol and UpgradeRewardsDistributorV2Test.t.sol, that covered protocols public functions, however the majority of tests located in HourglassProtocolTest were not updated to use the v2 contracts. We recommend updating these tests!



```
| File                                                           | % Lines          | % Statements     | % Branches      | % Funcs         |
|----------------------------------------------------------------|------------------|------------------|-----------------|-----------------|
| FeeManager.sol                                                 | 60.00% (12/20)   | 60.00% (12/20)   | 83.33% (5/6)    | 80.00% (4/5)    |
| HourglassCustodian.sol                                         | 76.80% (96/125)  | 65.13% (99/152)  | 56.67% (34/60)  | 71.43% (20/28)  |
| HourglassCustodianV2.sol                                       | 44.55% (45/101)  | 38.21% (47/123)  | 30.00% (15/50)  | 39.13% (9/23)   |
| RewardsDistributor.sol                                         | 42.22% (19/45)   | 42.62% (26/61)   | 37.50% (3/8)    | 63.64% (7/11)   |
| sRewardsDistributorV2.sol                                      | 0.00% (0/37)     | 0.00% (0/53)     | 0.00% (0/8)     | 0.00% (0/11)    |
| eth-vaults/convex-frax-asset/ConvexFraxMaturedHoldings.sol     | 0.00% (0/35)     | 0.00% (0/41)     | 0.00% (0/12)    | 0.00% (0/12)    |
| eth-vaults/convex-frax-asset/ConvexFraxVault.sol               | 91.38% (53/58)   | 92.19% (59/64)   | 50.00% (11/22)  | 84.62% (11/13)  |
| eth-vaults/convex-frax-asset/HourglassConvexFraxReceipt.sol    | 68.89% (31/45)   | 66.07% (37/56)   | 37.50% (9/24)   | 40.00% (6/15)   |
| eth-vaults/convex-frax-asset/HourglassConvexFraxReceiptV2.sol  | 0.00% (0/40)     | 0.00% (0/53)     | 0.00% (0/20)    | 0.00% (0/13)    |
| eth-vaults/general/ERC20Intermediary.sol                       | 100.00% (5/5)    | 100.00% (5/5)    | 100.00% (2/2)   | 100.00% (5/5)   |
| eth-vaults/general/GeneralMatureHoldingVault.sol               | 83.33% (10/12)   | 83.33% (10/12)   | 75.00% (3/4)    | 72.73% (8/11)   |
| eth-vaults/general/GeneralMatureHoldingVaultV2.sol             | 40.00% (6/15)    | 40.00% (6/15)    | 25.00% (1/4)    | 33.33% (4/12)   |
| eth-vaults/general/TwoStepOwnable.sol                          | 0.00% (0/20)     | 0.00% (0/20)     | 0.00% (0/8)     | 0.00% (0/7)     |
| Burn.sol                                                       | 100.00% (3/3)    | 100.00% (3/3)    | 50.00% (1/2)    | 100.00% (1/1)   |
| HourglassToken.sol                                             | 100.00% (9/9)    | 100.00% (10/10)  | 100.00% (4/4)   | 100.00% (4/4)   |
```

Static analysis

During our audit of the Hourglass smart contracts, we conducted a thorough static code analysis using a set of our proprietary rules. This analysis aimed to identify any known bugs or recurring attack vectors. The results revealed that all issues detected were false positives, indicating the absence of actual vulnerabilities or bugs. Our automated examination applied specific rules designed to detect common coding mistakes, security vulnerabilities, and attack patterns. We carefully verified each reported issue and determined that they did not pose real security risks. This static analysis provides additional assurance of the codebase's thorough examination and reinforces confidence in the security and reliability of the Hourglass smart contracts. For more detailed information, please refer to the comprehensive report provided. In conclusion, the static code analysis confirmed the absence of genuine vulnerabilities or bugs, further enhancing the security assessment's credibility.



```
Scanning 25 files with 42 solidity rules.
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 25/25 tasks 0:00:00

 ┌─────────┐
 │ Results │
 └─────────┘

Findings:

  /hourglass-platform/src/ethereum/eth-vaults/convex-frax-asset/ConvexFraxVault.sol
         solidity.basic-arithmetic-underflow
         Possible arithmetic underflow

              bytes32 kekId = IConvexVault(depositVault).stakeLockedCurveLp(initAmount, (_maturityTimestamp - block.timestamp));


  /hourglass-platform/src/ethereum/eth-vaults/convex-frax-asset/HourglassConvexFraxReceipt.sol
         solidity.erc20-public-burn

              function burn(address account, uint256 id, uint256 value) public virtual {


 ┌──────────────┐
 │ Scan Summary │
 └──────────────┘

Ran 42 rules on 25 files: 2 findings.
```

Integration with Other Protocols

In the process of auditing the Hourglass contracts, we have identified that the project directly or indirectly interfaces with, or is based upon, a list of existing protocols. These protocols include Curve Finance, Convex Finance, Votium, and Frax Finance. The integration of multiple protocols within a project brings both opportunities and potential risks. This summary highlights the significance integrating with other protocols has for protocol security and emphasizes the limitations of the audit scope.

Enhancing Interoperability:

Integrating with well-established protocols enables the project to leverage existing functionalities and tap into a broader ecosystem. This interoperability provides numerous benefits, including enhanced liquidity, access to additional services, and the ability to leverage established communities. By integrating with these protocols, the project positions itself to deliver a more comprehensive and seamless experience for its users. At the same time, each integration brings another layer of complexity into the project and makes it more difficult for the team to ensure project security.

Risk Assessment:

During our audit, we diligently analyzed the Hourglass contracts for known issues or common mistakes that may arise when interacting with the aforementioned protocols. Our objective was to identify any potential vulnerabilities or risks within the contracts that could impact the integration. However, it is important to note that our audit scope was specifically limited to the Hourglass contracts and did not encompass a comprehensive verification of the integrated protocols themselves.

Ongoing Risk Management:

To mitigate potential risks associated with protocol integrations, it is imperative for the project to maintain a proactive approach to risk management. This includes staying informed about updates, upgrades, and potential vulnerabilities of the integrated protocols. By actively monitoring the security landscape and engaging in ongoing risk assessments, the project can promptly address any emerging threats or vulnerabilities, safeguarding the interests of its users and the overall integrity of the system.

The integration of the project with other protocols, including Curve Finance, Convex Finance, and Frax Finance, expands its capabilities and potential. While we have analyzed the Hourglass contracts for issues related to these integrated protocols, it is important to reiterate that our audit scope did not encompass the verification of the integrated protocols themselves. Therefore, it is crucial for project stakeholders to undertake independent audits of the integrated protocols to ensure their security and reliability. By adopting a proactive approach to risk management and ongoing assessments, the project can effectively navigate the complexities of integration and deliver a secure and robust experience for its users.