Mimo DeFi - Parallel Protocol
This security assessment was prepared by Quantstamp, the leader in blockchain security.

Executive Summary

Type: Decentralized Stablecoin
Auditors: Kacper Bąk, Senior Research Engineer
Kevin Feng, Blockchain Researcher
Jan Gorzny, Blockchain Researcher
Timeline: 2020-11-02 through 2020-11-19
EVM: Muir Glacier
Languages: Solidity, Typescript
Specification: Whitepaper

Documentation Quality: High
Test Quality: High
Source Code: Repository: titan Commit: 680fa1a

Goals:
- Can funds get locked in the contract?
- Can user's collateral get liquidated prematurely?

Total Issues: 8 (4 Resolved)
High Risk Issues: 0 (0 Resolved)
Medium Risk Issues: 0 (0 Resolved)
Low Risk Issues: 6 (3 Resolved)
Informational Risk Issues: 1 (1 Resolved)
Undetermined Risk Issues: 1 (0 Resolved)

Information:
- High Risk: The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- Medium Risk: The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
- Low Risk: The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low-impact in view of the client's business circumstances.
- Informational: The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
- Undetermined: The impact of the issue is uncertain.

- Unresolved: Acknowledged the existence of the risk, and decided to accept it without engaging in special efforts to control it.
- Acknowledged: The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).
- Resolved: Adjusted program implementation, requirements or constraints to eliminate the risk.
- Mitigated: Implemented actions to minimize the impact or likelihood of the risk.
Summary of Findings

We have found a few low-severity issues and two issues of undetermined severity. Overall, the code is well-written, well-documented, and well-tested. We recommend addressing all the issues to ensure code correctness with respect to the intent and to ensure it follows best practices.

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Severity</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>QSP-1</td>
<td>Privileged Roles and Ownership</td>
<td>Low</td>
<td>Acknowledged</td>
</tr>
<tr>
<td>QSP-2</td>
<td>Dependency on external contracts</td>
<td>Low</td>
<td>Acknowledged</td>
</tr>
<tr>
<td>QSP-3</td>
<td>Liquidation may be impossible if there is not funds in the insurance fund</td>
<td>Low</td>
<td>Acknowledged</td>
</tr>
<tr>
<td>QSP-4</td>
<td>Various calls to ERC20 functions are not in require statements</td>
<td>Low</td>
<td>Fixed</td>
</tr>
<tr>
<td>QSP-5</td>
<td>Lack of input validation</td>
<td>Low</td>
<td>Fixed</td>
</tr>
<tr>
<td>QSP-6</td>
<td>Potential division by zero</td>
<td>Low</td>
<td>Fixed</td>
</tr>
<tr>
<td>QSP-7</td>
<td>Unlocked Pragma</td>
<td>Informational</td>
<td>Fixed</td>
</tr>
<tr>
<td>QSP-8</td>
<td>Potential sensitivity to price movements</td>
<td>Undetermined</td>
<td>Acknowledged</td>
</tr>
</tbody>
</table>

Quantstamp Audit Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

The Quantstamp auditing process follows a routine series of steps:

1. Code review that includes the following
   i. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.

2. Testing and automated analysis that includes the following:
   i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
   ii. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.

3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Toolset

The notes below outline the setup and steps performed in the process of this audit.

Setup

Tool Setup:

- `slither v0.6.12`

Steps taken to run the tools:

1. Installed the Slither tool: `pip install slither-analyzer`
2. Run Slither from the project directory: `slither`
Findings

QSP-1 Privileged Roles and Ownership

Severity: Low Risk
Status: Acknowledged

File(s) affected: EURX.sol, USDX.sol, FeeDistributor.sol, AddressProvider.sol, ConfigProvider.sol, PriceFeed.sol, PriceFeedEUR.sol

Description: Smart contracts will often have owner variables to designate the person with special privileges to make modifications to the smart contract. Specifically, the contract EURX and USDX have a special minter role. The contracts FeeDistributor, AddressProvider, ConfigProvider, PriceFeed, PriceFeedEUR feature a manager role. Notably, in ConfigProvider.sol manager can set arbitrary debt limit, collateral ratio, borrow rate, etc.

Recommendation: This centralization of power needs to be made clear to the users, especially depending on the level of privilege the contract allows to the owner.
Update: The team informed us that they plan on decentralizing the governance before the full launch.

QSP-2 Dependency on external contracts

Severity: Low Risk
Status: Acknowledged

File(s) affected: PriceFeed.sol, PriceFeedEUR.sol, VaultsCore.sol

Description: The contracts PriceFeed, PriceFeedEUR, and VaultsCore depend on external contracts. Furthermore, PriceFeed, PriceFeedEUR depend on decimals() to determine price accuracy. It is important to note that according to EIP-20, this method/field is optional. PriceFeedEUR depends on the transfer() functionality of external tokens.

Recommendation: We recommend vetting the underlying tokens carefully to ensure they are compatible with TenX contracts and that they behave as expected.

QSP-3 Liquidation may be impossible is there is not funds in the insurance fund

Severity: Low Risk
Status: Acknowledged

File(s) affected: VaultsCore.sol

Description: Due to the condition in L293 liquidation may be impossible if there is not enough funds in the insurance fund in a corner case when value of collateral drops below debt value.

Recommendation: The team has acknowledged this corner case scenario and informed us that they are planning on creating further fallback (e.g., automatic selling of governance token) once governance is in place. We do not have any further recommendations at this time.

QSP-4 Various calls to ERC20 functions are not in require statements

Severity: Low Risk
Status: Fixed

File(s) affected: VaultsCore.sol

Description: Calls to ERC20 functions on external assets are not wrapped in require() statements.

Recommendation: Since both functions return boolean values, we recommend wrapping them in require() statements.

QSP-5 Lack of input validation

Severity: Low Risk
Status: Fixed

File(s) affected: AddressProvider.sol, LiquidationManager.sol, ConfigProvider.sol, PriceFeed.sol, PriceFeedEUR.sol, RatesManager.sol, VaultsCore.sol, VaultsDataProvider.sol, FeeDistributor.sol, EURX.sol, USDX.sol

Description: Functions do not check if arguments of type address are non-zero in the following:

- AddressProvider.sol, functions: setAccessController(), setConfigProvider(), setVaultsCore(), setStableX(), setRatesManager(), setPriceFeed(), setVaultsDataProvider(), setFeeDistributor().
- ConfigProvider.sol, setCollateralConfig().
- LiquidationManager constructor().
- PriceFeed.setAssetOracle().
- PriceFeedEUR.sol, functions: setAssetOracle() and setEurOracle().
- RatesManager constructor().
- VaultsCore.sol, functions: constructor(), upgrade(), initializeRates(), refreshCollateral(), and deposit() (in contrast to a few below).
- VaultsDataProvider.sol, functions: constructor() and createVault() (in contrast to a few below).
- FeeDistributor constructor().
- EURX constructor().
- USDX constructor().

Recommendation: We recommend adding the relevant checks.

QSP-6 Potential division by zero

Severity: Low Risk
Status: Fixed
There is a chance that `minRatio` is set to 0. In case that happens, the division in `LiquidationManager.sol` would fail due to division by 0.

**QSP-7 Unlocked Pragma**

**Severity:** Informational  
**Status:** Fixed

The code is well documented.

**QSP-8 Potential sensitivity to price movements**

**Severity:** Undetermined  
**Status:** Acknowledged

Adherence to Best Practices

1. In `RatesManager.sol`, the modifier `onlyVaultsCore` is unused. **Update:** fixed.

2. In `FeeDistributor.sol` and `IConfigProvider.sol`, TODO items. **Update:** fixed.

Test Results

The project features a comprehensive test suite.
gasUsed withdraw: 226953

gasUsed withdraw: 55065

gasUsed deposit: 230557

32 pending

Contract: VaultsCore vaults

Contract: VaultsCore EUR vaults

Contract: VaultsCore rates

Contract: VaultsCore rates multi collateral

Contract: VaultsCore income

Contract: VaultsCore GAS costs

Contract: VaultsCore Debt

Contract: VaultsCore Debt Limits

Contract: RatesManager calculations

✓ upgrading to new vaultscore should approve tokentranfers for both stablex and collateral types (92ms)

✓ vault owner cannot borrow and let the health factor go below 1 (628ms)

✓ vault owner can repay without specifying amount (448ms)

✓ vault owner can repay fully (438ms)

✓ vault owner can repay partially (454ms)

✓ should calculate income correctly with orignation fee

✓ should calculate origination fee correctly after time has passed (518ms)

✓ vault owner can borrow (343ms)

✓ vault owner can withdraw part of his vault collateral (193ms)

✓ vault owner cannot withdraw more than vault balance (185ms)

✓ depositing with sufficient allowance should add to vault balance (153ms)

✓ depositing without sufficient allowance should fail (138ms)

✓ user can open a vault via a deposit (122ms)

✓ vault owner cannot withdraw and let the health factor go below 1 (582ms)

✓ setBorrowRate should update income (500ms)

✓ Cumulative Rate should change as time passes (155ms)

✓ should be able to set a new rate (84ms)

✓ should initialize the last rate update timestamp to deployment timestamp

✓ should initialize cumulative and current rate to 1

- should calculate income correctly when borrowing more

- should calculate income correctly when liquidating

- should calculate income correctly when repaying

- should calculate income correctly with changing interest rates

- should calculate income correctly for multiple vaults

✓ withdraw should update income and cumulative rate (712ms)

✓ should add origination fee to income when borrowing (351ms)

✓ GAS for repayment should not exceed 260k GAS (414ms)

✓ GAS for borrowing should not exceed 380k GAS (328ms)

✓ GAS for withdrawing collateral from a vault with debt should not exceed 280k GAS (397ms)

✓ GAS for withdrawing collateral from no debt vault should not exceed 180k GAS (176ms)

✓ GAS for depositing collateral should not exceed 240k GAS (132ms)

- should calculate income correctly when repaying

- should calculate income correctly with changing interest rates

- should calculate income correctly for multiple vaults

✓ withdraw should update income and cumulative rate (712ms)

✓ should add origination fee to income when borrowing (351ms)

✓ GAS for repayment should not exceed 260k GAS (414ms)

✓ GAS for withdrawing collateral should not exceed 280k GAS (397ms)

✓ GAS for depositing collateral should not exceed 240k GAS (132ms)

- should calculate income correctly when repaying

- should calculate income correctly with changing interest rates

- should calculate income correctly for multiple vaults

✓ withdraw should update income and cumulative rate (712ms)

✓ should add origination fee to income when borrowing (351ms)

✓ GAS for repayment should not exceed 260k GAS (414ms)

✓ GAS for withdrawing collateral should not exceed 280k GAS (397ms)

✓ GAS for depositing collateral should not exceed 240k GAS (132ms)

- should calculate income correctly when repaying

- should calculate income correctly with changing interest rates

- should calculate income correctly for multiple vaults

✓ withdraw should update income and cumulative rate (712ms)

✓ should add origination fee to income when borrowing (351ms)

✓ GAS for repayment should not exceed 260k GAS (414ms)

✓ GAS for withdrawing collateral should not exceed 280k GAS (397ms)

✓ GAS for depositing collateral should not exceed 240k GAS (132ms)

- should calculate income correctly when repaying

- should calculate income correctly with changing interest rates

- should calculate income correctly for multiple vaults

✓ withdraw should update income and cumulative rate (712ms)

✓ should add origination fee to income when borrowing (351ms)

✓ GAS for repayment should not exceed 260k GAS (414ms)

✓ GAS for withdrawing collateral should not exceed 280k GAS (397ms)

✓ GAS for depositing collateral should not exceed 240k GAS (132ms)

- should calculate income correctly when repaying

- should calculate income correctly with changing interest rates

- should calculate income correctly for multiple vaults

✓ withdraw should update income and cumulative rate (712ms)

✓ should add origination fee to income when borrowing (351ms)

✓ GAS for repayment should not exceed 260k GAS (414ms)

✓ GAS for withdrawing collateral should not exceed 280k GAS (397ms)

✓ GAS for depositing collateral should not exceed 240k GAS (132ms)

- should calculate income correctly when repaying

- should calculate income correctly with changing interest rates

- should calculate income correctly for multiple vaults

✓ withdraw should update income and cumulative rate (712ms)

✓ should add origination fee to income when borrowing (351ms)

✓ GAS for repayment should not exceed 260k GAS (414ms)

✓ GAS for withdrawing collateral should not exceed 280k GAS (397ms)

✓ GAS for depositing collateral should not exceed 240k GAS (132ms)

- should calculate income correctly when repaying

- should calculate income correctly with changing interest rates

- should calculate income correctly for multiple vaults

✓ withdraw should update income and cumulative rate (712ms)
## Code Coverage

The project has a good coverage. We recommend improving the coverage in `RatesManager.sol`.

<table>
<thead>
<tr>
<th>File</th>
<th>% Stmts</th>
<th>% Branch</th>
<th>% Funcs</th>
<th>% Lines</th>
<th>Uncovered Lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>access/</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>AccessController.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>chainlink/</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>AggregatorV3Interface.sol</td>
<td>94.72</td>
<td>81.94</td>
<td>95.35</td>
<td>94.53</td>
<td></td>
</tr>
<tr>
<td>core/</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>AddressProvider.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>ConfigProvider.sol</td>
<td>81.58</td>
<td>75</td>
<td>100</td>
<td>83.33</td>
<td>... 63, 64, 65, 66</td>
</tr>
<tr>
<td>LiquidityManager.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>PriceFeed.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>PriceFeedEUR.sol</td>
<td>95.05</td>
<td>87.5</td>
<td>100</td>
<td>96</td>
<td>54</td>
</tr>
<tr>
<td>RatesManager.sol</td>
<td>77.78</td>
<td>66.67</td>
<td>66.67</td>
<td>74, 75, 74</td>
<td></td>
</tr>
<tr>
<td>VaultsCore.sol</td>
<td>98.04</td>
<td>94.44</td>
<td>98.1</td>
<td>174, 175</td>
<td></td>
</tr>
<tr>
<td>VaultsDataProvider.sol</td>
<td>97.44</td>
<td>93.33</td>
<td>97.22</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>fees/</td>
<td>96.55</td>
<td>95.56</td>
<td>100</td>
<td>96.98</td>
<td>108</td>
</tr>
<tr>
<td>FeeDistributor.sol</td>
<td>96.55</td>
<td>95.56</td>
<td>100</td>
<td>96.98</td>
<td>108</td>
</tr>
<tr>
<td>interfaces/</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IAccessController.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IAddressProvider.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IConfigProvider.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IFeeDistributor.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>ILiquidityManager.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IPriceFeed.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IRatesManager.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IStableX.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IVaultsCore.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>IVaultsDataProvider.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>libraries/</td>
<td>78.26</td>
<td>66.67</td>
<td>78.26</td>
<td>78.26</td>
<td></td>
</tr>
<tr>
<td>MathPow.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>WadRayMath.sol</td>
<td>72.22</td>
<td>65.64</td>
<td>72.22</td>
<td>35, 57, 61, 67</td>
<td></td>
</tr>
<tr>
<td>mocks/</td>
<td>59.58</td>
<td>64.29</td>
<td>59.58</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MockChainlinkAggregator.sol</td>
<td>60</td>
<td>66.67</td>
<td>60</td>
<td>... 60, 61, 62, 63</td>
<td></td>
</tr>
<tr>
<td>MockChainlinkAggregatorEUR.sol</td>
<td>53.33</td>
<td>50</td>
<td>53.33</td>
<td>... 60, 61, 62, 63</td>
<td></td>
</tr>
<tr>
<td>MockWBTC.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>MockWETH.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>token/</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>EURX.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>USDRX.sol</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>All Files</td>
<td>90.42</td>
<td>78.57</td>
<td>89.94</td>
<td>90.73</td>
<td></td>
</tr>
</tbody>
</table>
The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

**Contracts**

680fa1a

**Tests**

86a69d8e1b352d5f9d69bc193287a297d7511777b9dd1da8e264594f3eba

---

**Appendix**

**File Signatures**

<table>
<thead>
<tr>
<th>Date</th>
<th>File Name</th>
<th>Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020-11-11</td>
<td>Initial report</td>
<td></td>
</tr>
<tr>
<td>2020-11-23</td>
<td>Revised report</td>
<td></td>
</tr>
</tbody>
</table>

---

**ChangeLog**

- 2020-11-11 – Initial report
- 2020-11-23 – Revised report based on commit 680fa1a
About Quantstamp

Quantstamp is a Y Combinator-backed company that helps to secure blockchain platforms at scale using computer-aided reasoning tools, with a mission to help boost the adoption of this exponentially growing technology.

With over 1000 Google scholar citations and numerous published papers, Quantstamp’s team has decades of combined experience in formal verification, static analysis, and software verification. Quantstamp has also developed a protocol to help smart contract developers and projects worldwide to perform cost-effective smart contract security scans.

To date, Quantstamp has protected $5B in digital asset risk from hackers and assisted dozens of blockchain projects globally through its white glove security assessment services. As an evangelist of the blockchain ecosystem, Quantstamp assists core infrastructure projects and leading community initiatives such as the Ethereum Community Fund to expedite the adoption of blockchain technology.

Quantstamp’s collaborations with leading academic institutions such as the National University of Singapore and MIT (Massachusetts Institute of Technology) reflect our commitment to research, development, and enabling world-class blockchain security.

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication.

Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

Links to other websites

You may, through hyperlinks or other computer links, gain access to web sites operated by persons other than Quantstamp, Inc. (Quantstamp). Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites’ owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on the websites and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.

Disclaimer

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Blockchain technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.