CERTIK

Security Assessment

Turbos Finance - Audit

CertiK Assessed on Sept 4th, 2023

CertiK Assessed on Sept 4th, 2023

Turbos Finance - Audit

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Sui	Manual Review, Static Analysis
LANGUAGE	**TIMELINE**	**KEY COMPONENTS**
Move	Delivered on 09/04/2023	N/A

CODEBASE

https://github.com/turbos-finance/turbos-clmm

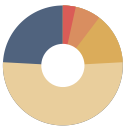
View All in Codebase Page

COMMITS

- 7ffa9e052cc3946198e7173aea3576e66fa7dbfc
- 56096599900e89cffbe4aa09dcbcfd749960df46
- d01e557e3fb613451b805c136f2a3f55d58d3d65

View All in Codebase Page

Vulnerability Summary



29	27	0	1	1	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined

1	Critical	1 Resolved	Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
2	Major	1 Resolved, 1 Acknowledged	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
4	Medium	4 Resolved	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
15	Minor	15 Resolved	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
7	Informational	6 Resolved, 1 Partially Resolved	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | TURBOS FINANCE - AUDIT

CODEBASE | TURBOS FINANCE - AUDIT

Repository

https://github.com/turbos-finance/turbos-clmm

Commit

- 7ffa9e052cc3946198e7173aea3576e66fa7dbfc
- 56096599900e89cffbe4aa09dcbcfd749960df46
- d01e557e3fb613451b805c136f2a3f55d58d3d65
- be79c9ee62833d358bb3f43fd6682d2f85af0435
- 46729df296524d16fade2c200abc446e51155b0e
- f8c716a17dabc351be6f8018112748d827fdcf14
- bad211723bb3f8a67e7919265627f2f3528f90fc
- 51dd68cd443afab6cec8af56f021e3ae0d555113
- 374a2aa97286eb90b8098c6fe52520e88f726229
- fec175b4b1b7a79f4d78e9aec4a274b19a2317ea

AUDIT SCOPE | TURBOS FINANCE - AUDIT

105 files audited ● 3 files with Acknowledged findings ● 1 file with Partially Resolved findings
● 17 files with Resolved findings ● 84 files without findings

ID	Repo	File	SHA256 Checksum
POU	turbos-finance/turbos-clmm	sources/pool.move	ef1b77d1f9fe259a5e42300293ce51bfa35e2c934d063b2bb78386aa242fde72
POF	turbos-finance/turbos-clmm	sources/pool_factory.move	9c2368537b6b0f7a6bd1b031df5a0667c1d763090f1ea692c283805e997b6f8a
REW	turbos-finance/turbos-clmm	sources/reward_manager.move	fd2b7fc414eff69e0b387f0b741f046b6d64a47c76a71b082762b0fc1af2f032
POT	turbos-finance/turbos-clmm	sources/position_manager.move	18fe09d11474e61c3e11cf3418b785268bf6ea9fdcb71a459ff6023feec6a5f7
I32	turbos-finance/turbos-clmm	sources/math/i32.move	4e135f61be37bdfdccb421043356d4201899c4a2a0640bdefc3c4a46f7f02c7f
FE1	turbos-finance/turbos-clmm	sources/fee10000bps.move	77899750aec501ab153c3c2ddb75e8598394c7c62ea88e0315d9b1b361420a0b
FE3	turbos-finance/turbos-clmm	sources/fee3000bps.move	5189638957cf8ca752850895d58b6fe868b74e4be7b64b7acbe05980f322257d
FE5	turbos-finance/turbos-clmm	sources/fee500bps.move	420453e64787be61760d03b1c327905740b143e0697619271b590ec8c7504303
MAS	turbos-finance/turbos-clmm	sources/math_sqrt_price.move	295d72c52551eda27a73323a0de826d4ac9896712d3347698b74ad3aefc8fa9e

ID	Repo	File	SHA256 Checksum
MAW	turbos-finance/turbos-clmm	sources/math_swap.move	1344ef7f5fa8e0f9a1020d6fe7ada4008d8ce4064b3f7a42d06cade23fed3f54
POO	turbos-finance/turbos-clmm	sources/pool.move	77c0721667e5fe4f5349491f3cf33f8deab7914b892ab10be45053b0f44cfbd0
POL	turbos-finance/turbos-clmm	sources/pool_factory.move	05a1953e9ba907442add616b0b5bb1251d2d3f4687e334a38adccc8cadf9714b
POS	turbos-finance/turbos-clmm	sources/position_manager.move	67a707b074b183df800d032620fe5368f9003d99af2de9443a464d7e00e4c74d
POI	turbos-finance/turbos-clmm	sources/position_nft.move	2f810605ee6e709190603f1ce76289e011aad001a394e7ebb78670982a13cb3b
STR	turbos-finance/turbos-clmm	sources/string_tools.move	51d488a0cd9a45fae6b2b90eefc1bec6e226cdb149b63904a752b25fbf62e4d6
SWA	turbos-finance/turbos-clmm	sources/swap_router.move	9cd12620b0c8b5eeafa9e118fc15d881756d8a42dcab39494fd59f17246d7622
SWP	turbos-finance/turbos-clmm	sources/swap_router.move	e720e7af243dae678f2eb5dc46ba993c6962262ce8bb882d7c6a23f5394bfa9d
POR	turbos-finance/turbos-clmm	sources/pool.move	73519fddafc05e0e50e6d1d1313856eeb7682b496e4053efd7d2fb6f147c6831
PO3	turbos-finance/turbos-clmm	sources/position_nft.move	067486edbeb61016ba514e717044922a659b1cd439fba68fc1c2c0f5ab4404dd
PO4	turbos-finance/turbos-clmm	sources/pool.move	6ee47269d0447e46b09df3ede63dc13da7a086cbeb56a52165421372339c720d

ID	Repo	File	SHA256 Checksum
SWO	turbos-finance/turbos-clmm	sources/swap_router.move	42ba24a70000cb97bada03de349dc1188c029f0ec024213b68b121a1d11f29bf
FUL	turbos-finance/turbos-clmm	sources/math/full_math_u128.move	df97d3c97cc19e1e80c6b18d89714283b3d5aa06f872ee94ee5c01e7701586fe
FUM	turbos-finance/turbos-clmm	sources/math/full_math_u32.move	df743447b1537b3add45a7d480ead4ee39a0848d767ed7f2f19259d21596c8ab
FUA	turbos-finance/turbos-clmm	sources/math/full_math_u64.move	82cc9e62b12d742159044aada4291b2e7a1038bf9ae2619abce04a55b49a7b0f
I12	turbos-finance/turbos-clmm	sources/math/i128.move	36748b8ad8f6fb9b1e982c91b2d3f503c5006462cfc7538668c623c972affdb7
I64	turbos-finance/turbos-clmm	sources/math/i64.move	abb4d0c6438d04a9658a93b24999bcc1f8f756ed60258af862a68f8d99de8c51
MAU	turbos-finance/turbos-clmm	sources/math/math_u128.move	315ebf9167714c1f1e1362282797d16e14377454af217f61ecb39b93c7881f33
MA2	turbos-finance/turbos-clmm	sources/math/math_u256.move	29c30939577a6e0efb190ae11c260b2921702a6c25884f02b5eb82e53c9d5c38
MA6	turbos-finance/turbos-clmm	sources/math/math_u64.move	e9681684788eb1c03652720942609b6d803def4c744316de39a48319a9170891
FEE	turbos-finance/turbos-clmm	sources/fee.move	c0384a5f6052ec2766c239d0a86fde370136d2197d16c5b9578118baf9b8c159
MAH	turbos-finance/turbos-clmm	sources/math_bit.move	af209b16de6b0f5a5d62f975c66e22fcbcbb738f3eddd27987b3d039973d21b9

ID	Repo	File	SHA256 Checksum
MAL	turbos-finance/turbos-clmm	sources/math_liquidity.move	be525fd99be4ff23162db02adb3cfff1b8b121f82b5ead680b259948bf19db7e
MAI	turbos-finance/turbos-clmm	sources/math_tick.move	ca1eac59c4bc80c3fbc7bf95de856981a2a8131f88d252817ba7ebd4a3e410f0
FUT	turbos-finance/turbos-clmm	sources/lib/full_math_u128.move	df97d3c97cc19e1e80c6b18d89714283b3d5aa06f872ee94ee5c01e7701586fe
FUH	turbos-finance/turbos-clmm	sources/lib/full_math_u32.move	df743447b1537b3add45a7d480ead4ee39a0848d767ed7f2f19259d21596c8ab
FUU	turbos-finance/turbos-clmm	sources/lib/full_math_u64.move	82cc9e62b12d742159044aada4291b2e7a1038bf9ae2619abce04a55b49a7b0f
I18	turbos-finance/turbos-clmm	sources/lib/i128.move	36748b8ad8f6fb9b1e982c91b2d3f503c5006462cfc7538668c623c972affdb7
I3L	turbos-finance/turbos-clmm	sources/lib/i32.move	294337b63de367365beea4d4884eb87e02654b73cd6bc8854db794318745ac74
I6L	turbos-finance/turbos-clmm	sources/lib/i64.move	abb4d0c6438d04a9658a93b24999bcc1f8f756ed60258af862a68f8d99de8c51
MAB	turbos-finance/turbos-clmm	sources/lib/math_bit.move	af209b16de6b0f5a5d62f975c66e22fcbcbb738f3eddd27987b3d039973d21b9
MAQ	turbos-finance/turbos-clmm	sources/lib/math_liquidity.move	e0b7e2a7729afdd068dc52665bcad42fa339d9b38916ba630e64e55f11e85a7a
MAR	turbos-finance/turbos-clmm	sources/lib/math_sqrt_price.move	2e37c50d23bf2902b4ee4554e0fff0944ca6ad131aaabaedfea4dd4a6627ed44

ID	Repo	File	SHA256 Checksum
MAA	turbos-finance/turbos-clmm	sources/lib/math_swap.move	aeafbdb5da86557631ab2231835b5cb026bb078ccd174edb34ae66afbe021822
MAC	turbos-finance/turbos-clmm	sources/lib/math_tick.move	ca1eac59c4bc80c3fbc7bf95de856981a2a8131f88d252817ba7ebd4a3e410f0
MA1	turbos-finance/turbos-clmm	sources/lib/math_u128.move	315ebf9167714c1f1e1362282797d16e14377454af217f61ecb39b93c7881f33
MA5	turbos-finance/turbos-clmm	sources/lib/math_u256.move	29c30939577a6e0efb190ae11c260b2921702a6c25884f02b5eb82e53c9d5c38
MA4	turbos-finance/turbos-clmm	sources/lib/math_u64.move	e9681684788eb1c03652720942609b6d803def4c744316de39a48319a9170891
STI	turbos-finance/turbos-clmm	sources/lib/string_tools.move	94530477c30ea99f17c4f7aa111167f8748ed6715ce9af0d4df48af6555da27b
FES	turbos-finance/turbos-clmm	sources/fee.move	3a06abcf72739b2a8b4e9c52a91287e4f6b879129c50eec0cee24368d13b176a
FE0	turbos-finance/turbos-clmm	sources/fee10000bps.move	541ab14a85112ab357d8f25850d11a0592d46f683375254bb697af03d34c062e
FEB	turbos-finance/turbos-clmm	sources/fee100bps.move	ece515e481f0a7b45b4d16324f261d75d71e3aa24b949e6ddc219666def9eda9
FEP	turbos-finance/turbos-clmm	sources/fee3000bps.move	dbaf368e956c5856bf51d367a8d97d795475ddaaba7d0b8ec5630297b4af6161
FEO	turbos-finance/turbos-clmm	sources/fee500bps.move	ced4034ca2be60e1a009903bdcda7a531b95e300c04c273e5db8d37e0cd730cb

ID	Repo	File	SHA256 Checksum
POE	turbos-finance/turbos-clmm	sources/pool_fetcher.move	83340f251cc99fb5c14a8284b1d58b82e92e835fd5273a34be965bcc3c16f7a5
PON	turbos-finance/turbos-clmm	sources/position_nft.move	8e8ae949050a483e1f762ab89c158adc74b2b534ca579d53fab8721aeff82533
FU1	turbos-finance/turbos-clmm	sources/lib/full_math_u128.move	df97d3c97cc19e1e80c6b18d89714283b3d5aa06f872ee94ee5c01e7701586fe
FU3	turbos-finance/turbos-clmm	sources/lib/full_math_u32.move	df743447b1537b3add45a7d480ead4ee39a0848d767ed7f2f19259d21596c8ab
FU6	turbos-finance/turbos-clmm	sources/lib/full_math_u64.move	82cc9e62b12d742159044aada4291b2e7a1038bf9ae2619abce04a55b49a7b0f
I1L	turbos-finance/turbos-clmm	sources/lib/i128.move	36748b8ad8f6fb9b1e982c91b2d3f503c5006462cfc7538668c623c972affdb7
I3I	turbos-finance/turbos-clmm	sources/lib/i32.move	294337b63de367365beea4d4884eb87e02654b73cd6bc8854db794318745ac74
I6I	turbos-finance/turbos-clmm	sources/lib/i64.move	abb4d0c6438d04a9658a93b24999bcc1f8f756ed60258af862a68f8d99de8c51
MAO	turbos-finance/turbos-clmm	sources/lib/math_bit.move	af209b16de6b0f5a5d62f975c66e22fcbcbb738f3eddd27987b3d039973d21b9
MAD	turbos-finance/turbos-clmm	sources/lib/math_liquidity.move	e0b7e2a7729afdd068dc52665bcad42fa339d9b38916ba630e64e55f11e85a7a
MAP	turbos-finance/turbos-clmm	sources/lib/math_sqrt_price.move	2e37c50d23bf2902b4ee4554e0fff0944ca6ad131aaabaedfea4dd4a6627ed44

ID	Repo	File	SHA256 Checksum
MAE	turbos-finance/turbos-clmm	sources/lib/math_swap.move	aeafbdb5da86557631ab2231835b5cb026bb078ccd174edb34ae66afbe021822
MAK	turbos-finance/turbos-clmm	sources/lib/math_tick.move	ca1eac59c4bc80c3fbc7bf95de856981a2a8131f88d252817ba7ebd4a3e410f0
MA8	turbos-finance/turbos-clmm	sources/lib/math_u128.move	315ebf9167714c1f1e1362282797d16e14377454af217f61ecb39b93c7881f33
MA3	turbos-finance/turbos-clmm	sources/lib/math_u256.move	29c30939577a6e0efb190ae11c260b2921702a6c25884f02b5eb82e53c9d5c38
MA7	turbos-finance/turbos-clmm	sources/lib/math_u64.move	e9681684788eb1c03652720942609b6d803def4c744316de39a48319a9170891
STN	turbos-finance/turbos-clmm	sources/lib/string_tools.move	94530477c30ea99f17c4f7aa111167f8748ed6715ce9af0d4df48af6555da27b
FEU	turbos-finance/turbos-clmm	sources/fee.move	3a06abcf72739b2a8b4e9c52a91287e4f6b879129c50eec0cee24368d13b176a
FER	turbos-finance/turbos-clmm	sources/fee10000bps.move	541ab14a85112ab357d8f25850d11a0592d46f683375254bb697af03d34c062e
FEC	turbos-finance/turbos-clmm	sources/fee100bps.move	ece515e481f0a7b45b4d16324f261d75d71e3aa24b949e6ddc219666def9eda9
FE7	turbos-finance/turbos-clmm	sources/fee3000bps.move	dbaf368e956c5856bf51d367a8d97d795475ddaaba7d0b8ec5630297b4af6161
FE4	turbos-finance/turbos-clmm	sources/fee500bps.move	ced4034ca2be60e1a009903bdcda7a531b95e300c04c273e5db8d37e0cd730cb

ID	Repo	File	SHA256 Checksum
POA	turbos-finance/turbos-clmm	sources/pool_factory.move	478cc7b4fef724abc46b8e840a751f7865f16360dd26810593ed68850551dda8
POC	turbos-finance/turbos-clmm	sources/pool_fetcher.move	7daea8810deda5b9b54f7814bb93a981cab76d8848d33581aa7ca705e0248894
POM	turbos-finance/turbos-clmm	sources/position_manager.move	47c00bfe7fa4afc13cb101c4c60fd3b5dc68d5d580dd6ae2bf0105c8cf531816
REA	turbos-finance/turbos-clmm	sources/reward_manager.move	d5912c2fb06dc9f80359e087e6beda906624c8bff6ec0f3b34d05b2594faf5c3
SWR	turbos-finance/turbos-clmm	sources/swap_router.move	42ba24a70000cb97bada03de349dc1188c029f0ec024213b68b121a1d11f29bf
FU2	turbos-finance/turbos-clmm	sources/lib/full_math_u128.move	df97d3c97cc19e1e80c6b18d89714283b3d5aa06f872ee94ee5c01e7701586fe
FUI	turbos-finance/turbos-clmm	sources/lib/full_math_u32.move	df743447b1537b3add45a7d480ead4ee39a0848d767ed7f2f19259d21596c8ab
FU4	turbos-finance/turbos-clmm	sources/lib/full_math_u64.move	82cc9e62b12d742159044aada4291b2e7a1038bf9ae2619abce04a55b49a7b0f
I1I	turbos-finance/turbos-clmm	sources/lib/i128.move	36748b8ad8f6fb9b1e982c91b2d3f503c5006462cfc7538668c623c972affdb7
I3B	turbos-finance/turbos-clmm	sources/lib/i32.move	294337b63de367365beea4d4884eb87e02654b73cd6bc8854db794318745ac74
I6B	turbos-finance/turbos-clmm	sources/lib/i64.move	abb4d0c6438d04a9658a93b24999bcc1f8f756ed60258af862a68f8d99de8c51

ID	Repo	File	SHA256 Checksum
MA9	turbos-finance/turbos-clmm	sources/lib/math_bit.move	af209b16de6b0f5a5d62f975c66e22fcbcbb738f3eddd27987b3d039973d21b9
MAY	turbos-finance/turbos-clmm	sources/lib/math_liquidity.move	e0b7e2a7729afdd068dc52665bcad42fa339d9b38916ba630e64e55f11e85a7a
MA0	turbos-finance/turbos-clmm	sources/lib/math_sqrt_price.move	7ed9be5345494c00cb4ed1de071e3c7e46b303d8fac817250129a6b4f5930242
MAF	turbos-finance/turbos-clmm	sources/lib/math_swap.move	aeafbdb5da86557631ab2231835b5cb026bb078ccd174edb34ae66afbe021822
MAM	turbos-finance/turbos-clmm	sources/lib/math_tick.move	ca1eac59c4bc80c3fbc7bf95de856981a2a8131f88d252817ba7ebd4a3e410f0
MAN	turbos-finance/turbos-clmm	sources/lib/math_u128.move	315ebf9167714c1f1e1362282797d16e14377454af217f61ecb39b93c7881f33
MAG	turbos-finance/turbos-clmm	sources/lib/math_u256.move	29c30939577a6e0efb190ae11c260b2921702a6c25884f02b5eb82e53c9d5c38
MTH	turbos-finance/turbos-clmm	sources/lib/math_u64.move	e9681684788eb1c03652720942609b6d803def4c744316de39a48319a9170891
STG	turbos-finance/turbos-clmm	sources/lib/string_tools.move	94530477c30ea99f17c4f7aa111167f8748ed6715ce9af0d4df48af6555da27b
FED	turbos-finance/turbos-clmm	sources/fee.move	3a06abcf72739b2a8b4e9c52a91287e4f6b879129c50eec0cee24368d13b176a
FE9	turbos-finance/turbos-clmm	sources/fee10000bps.move	541ab14a85112ab357d8f25850d11a0592d46f683375254bb697af03d34c062e

ID	Repo	File	SHA256 Checksum
FE8	turbos-finance/turbos-clmm	sources/fee100bps.move	ece515e481f0a7b45b4d16324f261d75d71e3aa24b949e6ddc219666def9eda9
FE2	turbos-finance/turbos-clmm	sources/fee3000bps.move	dbaf368e956c5856bf51d367a8d97d795475ddaaba7d0b8ec5630297b4af6161
FE6	turbos-finance/turbos-clmm	sources/fee500bps.move	ced4034ca2be60e1a009903bdcda7a531b95e300c04c273e5db8d37e0cd730cb
POY	turbos-finance/turbos-clmm	sources/pool_factory.move	7688a18759bfcf60b0fdc52b4ed8cc92acef4b2860d88b8b3950a499ecb973f7
POH	turbos-finance/turbos-clmm	sources/pool_fetcher.move	7daea8810deda5b9b54f7814bb93a981cab76d8848d33581aa7ca705e0248894
POG	turbos-finance/turbos-clmm	sources/position_manager.move	e2fd43d1b8b918ad9a5bb0cb55b543aae9bae3cce7d6133ac07c07cbfd336a1e
POD	turbos-finance/turbos-clmm	sources/position_nft.move	c84a215f8dae97e725df8624a11c416386b47b2eed568ea71fe5f24e99950558
RER	turbos-finance/turbos-clmm	sources/reward_manager.move	d5912c2fb06dc9f80359e087e6beda906624c8bff6ec0f3b34d05b2594faf5c3

APPROACH & METHODS | TURBOS FINANCE - AUDIT

This report has been prepared for Turbos Finance to discover issues and vulnerabilities in the source code of the Turbos Finance - Audit project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Manual Review and Static Analysis techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Testing the smart contracts against both common and uncommon attack vectors;
- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

REVIEW NOTES | TURBOS FINANCE - AUDIT

Overview

Turbos has implemented a concentrated liquidity market maker on Sui with the Move language. This infrastructure supports decentralized finance, allowing developers, traders, and liquidity providers to participate easily in the financial market.

The following features are provided in Turbos:

Automated Market Making

Turbos Finance serves as an automated market-making platform designed to enable the exchange of token pairs and liquidity provision.

Concentrated Liquidity

Turbos Finance allows for positions that concentrate liquidity within customized price ranges, enabling the provision of larger amounts of liquidity within desired price ranges. This boosts liquidity efficiency, resulting in a more stable price and less slippage within specific price ranges.

External Dependencies

The project is developed using Move language and running on the top of the Sui blockchain. The vulnerability and the updates of the language/Sui codebase might affect the project as a whole. As the Sui network is rapidly evolving, to avoid any potential compatibility issues and take advantage of new features and improvements, it's advisable for the client to upgrade the Sui framework to the most recent version. Additionally, staying informed about any upcoming updates or changes to the language or framework can help ensure the project remains secure and compatible.

The above dependencies are not within the current audit scope and serve as a black box. Modules/Contracts within the module are assumed to be valid and non-vulnerable actors in this audit and implement proper logic to collaborate with the current project and other modules.

Privileged Roles

To set up the project correctly, improve overall project quality and preserve upgradability, the following roles present in the audit(more details in **SOR-02: Centralization Risk**):

- `PoolFactoryAdmin` : for fee setup in the pool and NFT setting.
- `RewardManagerAdmin` : for reward distribution.

The advantage of the privileged role in the codebase is that the client reserves the ability to adjust the protocol according to the runtime required to serve the community best. It is also worthy of note the potential drawbacks of these functions, which should be clearly stated through the client's action/plan. Additionally, if the key pair of privileged accounts are compromised, the project could have devastating consequences.

To improve the trustworthiness of the project, dynamic runtime updates in the project should be notified to the community. Furthermore, any plan to invoke the aforementioned functions should also be considered to move to the execution queue of the `Timelock` contract.

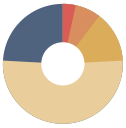
Upgradebility

Developers working with the Sui blockchain have the ability to upgrade packages based on their software iteration requirements. However, this also means that the `UpgradeCap` and publisher's key store should be handled with caution to prevent any unexpected loss. Additionally, it's important to inform the community about any upgrade plans to address concerns related to centralization and ensure transparency.

More information can be found:

- Sui Package Upgrades
- Third-Party Package upgrades

FINDINGS | TURBOS FINANCE - AUDIT



29	1	2	4	15	7
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Turbos Finance - Audit. Through this audit, we have uncovered 29 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
SWO-01	Missing Validation On Multi-Swap	Logical Issue	Critical	Resolved
SOR-02	**Centralization Risk**	**Centralization**	**Major**	**Acknowledged**
SWP-01	Incorrect `a_to_b` Values For Multiple Swaps	Logical Issue	Major	Resolved
MAS-01	Incorrect Square Root Price From Input Calculation	Logical Issue	Medium	Resolved
POR-01	Lack Of Check On Vault	Logical Issue	Medium	Resolved
POT-01	Insufficient Burn Condition	Logical Issue	Medium	Resolved
SOR-01	Missing Corresponding Functionality To Unlock The Pool	Logical Issue	Medium	Resolved
I32-01	Incorrect Euclidean Modulo For Negative Divisors	Incorrect Calculation	Minor	Resolved
MOV-01	Incompatibility Sui-Framework Location For Newer Version	Logical Issue	Minor	Resolved
PO3-01	Incorrect Return Value Of `position_id()`	Inconsistency	Minor	Resolved
PO4-01	Incorrect Reward Values Used	Logical Issue	Minor	Resolved

ID	Title	Category	Severity	Status
PO4-02	Missing Cases When Modifying Ticks	Logical Issue	Minor	Resolved
POE-01	Missing Version Check On Flash Swap Functions	Inconsistency	Minor	Resolved
POI-01	`Burn` Should Be `friend` Only	Logical Issue	Minor	Resolved
POL-01	Missing Input Validation On Token Type When Generating The Pool	Logical Issue	Minor	Resolved
POR-02	Incorrect Return Value In `next_pool_reward_infos()`	Logical Issue	Minor	Resolved
POU-01	Incorrect Condition For Swaps	Logical Issue	Minor	Resolved
SOU-01	Incorrect Transfer Functions On Objects Not Defined In Modules	Coding Issue	Minor	Resolved
SOU-02	Inconsistent Fee Limit	Inconsistency	Minor	Resolved
STR-01	Zero Converts To The Empty String	Logical Issue	Minor	Resolved
SWA-01	Missing Validation On `_deadline`	Logical Issue	Minor	Resolved
SWA-02	Unused Variable And Potential Missing Validation	Logical Issue	Minor	Resolved
GLOBAL-01	Missing Emit Events	Coding Style	Informational	Resolved
GLOBAL-02	Tick Modification	Inconsistency	Informational	Resolved
PO4-03	Invalid Assignment Issue	Coding Style	Informational	Resolved
POF-02	Test Only Function	Coding Issue	Informational	Resolved

ID	Title	Category	Severity	Status
POL-02	Possible To Deploy Duplicate Pools	Logical Issue	Informational	Resolved
POT-02	Reward Distribution	Logical Issue	Informational	Partially Resolved
SWA-03	Missing Validation On Empty Vector	Logical Issue	Informational	Resolved

SWO-01 | MISSING VALIDATION ON MULTI-SWAP

Category	Severity	Location	Status
Logical Issue	Critical	sources/swap_router.move (06/12-reward-fix): 169	Resolved

Description

Contract commit: 43d97c84923065996d31758c4ea858096b92f331 File: `swap_router.move`

The module `swap_router` is working as an interface to help users access the swap service. It mainly provides two kinds of swap:

1. single swap, for example, from SUI to TURBOS.
2. multi swap, for example, from SUI to TURBOS, then from TURBOS to USDC.

The single swap will invoke `pool::swap` to calculate the swap result, which updates the pool, and then invoke `pool::swap_coin` to process the coin change.

For multiswap, it will invoke `pool::swap` twice. Taking the exact out mode as an example:

1. swaps `step2_in` token y for `step2_out` token z
2. swaps `step1_in` token x for `step1_out` token y

Then the function `pool::swap_coin` processes the balance changes.

However, in the scenario that a pool does not have enough liquidity, which leads to `step1_out` possibly not equalling `step2_in`, an imbalanced swap can be generated.

Recommendation

We recommend the team adding validations on the `pool::swap` result to ensure the swap result is consistent during the multi swap methods.

Alleviation

[**Turbos Finance**, 07/31/2023]: The team heeded the advice and resolved the issue in commit bad211723bb3f8a67e7919265627f2f3528f90fc by adding validation on swap results inside the multi swap methods.

The team has also implemented a price protection mechanism on the frontend such that if a certain swap with execution price off from the market price on CEXs by more than 10% (1% for stable-stable), it will not be allowed.

[**CertiK**, 07/31/2023]: Depending solely on front-end protection is insufficient for securing an on-chain contract, as attackers have the ability to interact directly with the on-chain code.

SOR-02 | CENTRALIZATION RISK

Category	Severity	Location	Status
Centralization	Major	sources/pool.move (04/27-5609659): 661, 685, 721; sources/pool_factory.move (04/27-5609659): 177, 194, 204, 221, 229, 241, 253; sources/reward_manager.move (04/27-5609659): 22	Acknowledged

Description

In the project "Turbo-Clmm", the privileged capability `PoolFactoryAdminCap`, which is assigned to the contract deployer, has authority over the following functions:

- `pool_factory::set_fee_tier`: enable a fee amount with the given tick spacing
- `pool_factory::set_fee_protocol`: set up protocol fee
- `pool_factory::collect_protocol_fee`: collect protocol fee
- `pool_factory::toggle_pool_status`: disable/enable the pool
- `pool_factory::update_nft_name`: update the name of position nft
- `pool_factory::update_nft_description`: update the description of position nft
- `pool_factory::update_nft_img_url`: update the image URL of position nft
- `pool_factory::upgrade`: upgrade the protocol version
- `pool_factory::migrate_position`: migrate pool position to v2

Any compromise to the account with the `PoolFactoryAdminCap` capability may allow the attacker to manipulate the project.

The privileged capability `RewardManagerAdminCap`, which is assigned to the contract deployer as reward manager, has authority over the following functions:

- `reward_manager::init_reward`: init the new reward plan to the pool
- `reward_manager::update_reward_manager`: change the reward manager

Any compromise to the reward manager account may allow the attacker to manipulate the project.

Also, developers working with the Sui blockchain can upgrade packages based on their software iteration requirements. However, this also means that the `UpgradeCap` and deployer's key store should be handled cautiously to prevent any unexpected loss. Additionally, it's important to inform the community about any upgrade plans to address concerns related to centralization and ensure transparency.

More information can be found:

- Sui Package Upgrades

- Third-Party Package upgrades

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We recommend carefully managing the privileged account's keystore to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement;
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles;
  OR
- Remove the risky functionality.

Alleviation

[**Turbos Finance**, 05/01/2023]: The team acknowledged the finding and plan to cooperate with Msafe to manage the related `ManagerCap` using multi-signature addresses. There is no production release of Msafe available on Sui yet. It is expected to be available in 2-3 weeks.

SWP-01 | INCORRECT `a_to_b` VALUES FOR MULTIPLE SWAPS

Category	Severity	Location	Status
Logical Issue	Major	sources/swap_router.move (04/27-5609659): 166~167, 448~449	Resolved

Description

In the function `swap_a_b_b_c()`, two swaps are performed: `tokenA` to `tokenB` and `tokenB` to `tokenC`. However, when the swap is performed expecting an exact out amount of tokens, the swap is done backwards.

```
166             let a_to_b_step_one = false;
167             let a_to_b_step_two = false;
168             let (step2_out, step2_in) = pool::swap(
169                 pool_b,
170                 recipient,
171                 a_to_b_step_two,
172                 (amount as u128),
173                 is_exact_in,
174                 sqrt_price_limit_two,
175                 clock,
176                 ctx
177             );
178
179             let (step1_out, step1_in) = pool::swap(
180                 pool_a,
181                 recipient,
182                 a_to_b_step_one,
183                 step2_in,
184                 is_exact_in,
185                 sqrt_price_limit_one,
186                 clock,
187                 ctx
188             );
```

In the first invocation of `pool::swap()`, we have `a_to_b_step_two == false` and `is_exact_in == false`. This means we are swapping `tokenC` for `tokenB` expecting an exact amount out of `tokenB` tokens, when it should actually be `tokenC` tokens.

The result of this is that protocol fees, if any, are paid out in the wrong token, and there is a discrepancy in the amount of `tokenB` tokens that need to be put in for the swap. The below code snippet shows that protocol fees are always paid out by the input token, which should be `tokenB`, but due to the incorrect value of `a_to_b_step_two`, it will be paid out in `tokenC`.

```
            if (a_to_b) {
                pool.fee_growth_global_a = state.fee_growth_global;
                if (state.protocol_fee > 0) {
                    pool.protocol_fees_a = pool.protocol_fees_a + (state.protocol_fee as u64);
                };
            } else {
                pool.fee_growth_global_b = state.fee_growth_global;
                if (state.protocol_fee > 0) {
                    pool.protocol_fees_b = pool.protocol_fees_b + (state.protocol_fee as u64);
                };
            };
```

The same reasoning applies to the function `swap_a_b_b_c` in the `is_exact_in = false` branch.

Scenario

A scenario is provided to illustrate the above issue.

1. Protocol fees are set to 10%.
2. A multiswap is performed to send BTC -> USDC -> ETH.
3. Protocol fees are meant to be paid in BTC and USDC, but will actually be paid in USDC and ETH.

Proof of Concept

A test is provided to showcase the above scenario.

The function `prepare_tests_with_fee()` is the same as `prepare_tests()` except with the addition of the following line before pools are created:

```
        tools_tests::set_fee_protocol(admin, 100000, scenario);
```

The test:

```
    #[test]
    public fun test_incorrect_swap_abbc() {
        let admin = @0x0;
        let player = @0x1;
        let player2 = @0x2;

        let scenario_val = test_scenario::begin(admin);
        let scenario = &mut scenario_val;

        position_manager_tests::init_pool_manager(admin, scenario);

        prepare_tests_with_fee(admin, player, player2, scenario);

        let (pool_a_protocol_fee_a_before, pool_a_protocol_fee_a_after);
        let (pool_b_protocol_fee_a_before, pool_b_protocol_fee_a_after);
        let (pool_a_protocol_fee_b_before, pool_a_protocol_fee_b_after);
        let (pool_b_protocol_fee_b_before, pool_b_protocol_fee_b_after);

        test_scenario::next_tx(scenario, player);
        {
            let clock = test_scenario::take_shared<Clock>(scenario);
            let pool_a = test_scenario::take_shared<Pool<BTC, USDC, FEE3000BPS>>
(scenario);
            let pool_b = test_scenario::take_shared<Pool<USDC, ETH, FEE3000BPS>>
(scenario);

            //get trader balance before
            let coins_a;
            (coins_a, _) = tools_tests::get_user_coin<BTC>(scenario);

            (_, _, pool_a_protocol_fee_a_before, pool_a_protocol_fee_b_before, _, _,
_, _, _, _, _, _, _) = pool::get_pool_info(&pool_a);
            (_, _, pool_b_protocol_fee_a_before, pool_b_protocol_fee_b_before, _, _,
_, _, _, _, _, _, _) = pool::get_pool_info(&pool_b);

            swap_router::swap_a_b_b_c(
                &mut pool_a,
                &mut pool_b,
                coins_a,
                10000, //amount out
                10172, //amount_threshold
                MAX_SQRT_PRICE_X64 - 1,
                MAX_SQRT_PRICE_X64 - 1,
                false,
                player,
                1,
                &clock,
                test_scenario::ctx(scenario),
            );
```

```
            test_scenario::return_shared(clock);
            test_scenario::return_shared(pool_a);
            test_scenario::return_shared(pool_b);
        };

        test_scenario::next_tx(scenario, player);
        {
            let pool_a = test_scenario::take_shared<Pool<BTC, USDC, FEE3000BPS>>
(scenario);
            let pool_b = test_scenario::take_shared<Pool<USDC, ETH, FEE3000BPS>>
(scenario);

            (_, _, pool_a_protocol_fee_a_after, pool_a_protocol_fee_b_after, _, _,
_, _, _, _, _, _, _) = pool::get_pool_info(&pool_a);
            (_, _, pool_b_protocol_fee_a_after, pool_b_protocol_fee_b_after, _, _,
_, _, _, _, _, _, _) = pool::get_pool_info(&pool_b);

            // Fees are not paid using correspond A token
            assert_eq(pool_a_protocol_fee_a_after, pool_a_protocol_fee_a_before);
            assert_eq(pool_b_protocol_fee_a_after, pool_b_protocol_fee_a_before);

            // Fees are paid using corresponding B token
            assert!(pool_a_protocol_fee_b_after > pool_a_protocol_fee_b_before, 0);
            assert!(pool_b_protocol_fee_b_after > pool_b_protocol_fee_b_before, 1);

            test_scenario::return_shared(pool_a);
            test_scenario::return_shared(pool_b);
        };

        test_scenario::end(scenario_val);
    }
```

Recommendation

Recommend using the correct `a_to_b` values for multiple swaps. In fact, the `a_to_b` value should be independent of `is_exact_in`, as the direction of the swaps is always the same in both cases.

Note that the corresponding `step2_out, step2_in, step1_out, step1_in` variables also need to be changed as changing the `a_to_b` values changes which return value of `pool::swap()` corresponds to the exact in or exact out amounts.

```
        let (amount_a, amount_b) = if (a_to_b == amount_specified_is_input) {
            (amount_specified - state.amount_specified_remaining, state.
amount_calculated)
        } else {
            (state.amount_calculated, amount_specified - state.
amount_specified_remaining)
        };
```

Alleviation

[**Turbos Finance**, 05/01/2023]: The team heeded the advice and resolved the issue in commit 73ed4ed566fe7635793e20615ab4dd8e1c8f2209 by using the correct `a_to_b` values.

MAS-01 INCORRECT SQUARE ROOT PRICE FROM INPUT CALCULATION

Category	Severity	Location	Status
Logical Issue	Medium	sources/math_sqrt_price.move (04/19-7ffa9e0): 179, 181	Resolved

Description

The function `get_next_sqrt_price_from_input()` incorrectly removes the input token amount when it should instead add this amount. This causes prices to go in the opposite direction than intended.

```
168     public fun get_next_sqrt_price_from_input(
169         sqrt_price: u128,
170         liquidity: u128,
171         amount_in: u128,
172         a_for_b: bool
173     ): u128 {
174         assert!(sqrt_price > 0, EInvildSqrtPrice);
175         assert!(liquidity > 0, ELiquidity);
176
177         // round to make sure that we don't pass the target price
178         if (a_for_b) {
179             get_next_sqrt_price_from_amount_a_rounding_up(sqrt_price, liquidity
, amount_in, false)
180         } else {
181             get_next_sqrt_price_from_amount_b_rounding_down(sqrt_price,
 liquidity, amount_in, false)
182         }
183     }
```

The function calls `get_next_sqrt_price_from_amount_a_rounding_up()` or `get_next_sqrt_price_from_amount_b_rounding_down()` with `false` as the input for the parameter `add`.

This means that the `amount_in` value is removed from the pool, but for `get_next_sqrt_price_from_input()`, `amount_in` actually represents the value that is added to the pool. As a result, this issue causes prices to go in the opposite direction.

Although this function is not used within the project, it may be used by external users to calculate expected prices from various swap amounts. An incorrect calculation may affect the interoperability of the project.

Scenario

We illustrate a scenario showcasing the above issue:

1. The current price is queried.
2. 10 tokenA is then swapped into the pool.
3. The next price is queried.
4. The price should decrease as the amount of tokenA increased, but the price actually increases.

Proof of Concept

A unit test is provided to illustrate the above issue.

```
#[test]
fun test_incorrect_next_price() {
    let current_price = encode_price_sqrt(100, 100); // 100 tokenA, 100 tokenB
    let next_sqrt_price = get_next_sqrt_price_from_input(
        current_price,
        50, // liquidity
        10, // amount in
        true // a for b
    );

    // next price should be lower since the number of tokenA has increased
    assert!(next_sqrt_price > current_price, 0);
}
```

Recommendation

Recommend using `true` as the input for the parameter `add` in the function `get_next_sqrt_price_from_input()`.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 08d38c4837e9b4c432b2b1c0f264f073b67144d4 by removing the function `get_next_sqrt_price_from_input()`.

POR-01 | LACK OF CHECK ON VAULT

Category	Severity	Location	Status
Logical Issue	Medium	sources/pool.move (05/30-374a2aa): 918	Resolved

Description

The function `collect_reward_v2()` transfers rewards from a vault to the user. However, there are no checks to ensure that the provided vault object matches the provided pool object and reward index. This is inconsistent with the original `collect_reward()` function.

```
    public(friend) fun collect_reward_v2<CoinTypeA, CoinTypeB, FeeType, RewardCoin>(
        ...
    ): u64 {
        let position = get_position_mut(pool, position_owner, tick_lower_index,
tick_upper_index);
        assert!(reward_index <
vector::length(&position.reward_infos),EInvalidRewardIndex);
        let reward_info = vector::borrow_mut(&mut position.reward_infos,
reward_index);
```

```
    public(friend) fun collect_reward<CoinTypeA, CoinTypeB, FeeType, RewardCoin>(
        ...
    ): u64 {
        let owner = tx_context::sender(ctx);
        let pool_reward_info = vector::borrow(&pool.reward_infos, reward_index);
        assert!(pool_reward_info.vault == object::id_address(vault),
EInvalidRewardVault);
        let position = get_position_mut(pool, owner, tick_lower_index,
tick_upper_index);
        assert!(reward_index <
vector::length(&position.reward_infos),EInvalidRewardIndex);
        let reward_info = vector::borrow_mut(&mut position.reward_infos,
reward_index);
```

Due to this inconsistency, it is possible to use an unintended vault object, such as one for a different reward index or even one for a different pool. Depending on the value of each reward coin, users will always be able to claim rewards corresponding to the most profitable coin.

Recommendation

It is recommended to include checks to ensure that the correct vault object is used.

Alleviation

[**Turbos Finance**, 06/01/2023]: The team heeded the advice and resolved the issue in commit 51dd68cd443afab6cec8af56f021e3ae0d555113 by adding vault validation.

POT-01 | INSUFFICIENT BURN CONDITION

Category	Severity	Location	Status
Logical Issue	Medium	sources/position_manager.move (04/27-5609659): 186	Resolved

Description

When burning a position, a check is made to ensure that the position has no liquidity and tokens owed.

```
186         assert!(position.liquidity == 0 && position.tokens_owed_a == 0 &&
position.tokens_owed_b == 0, EPositionNotCleared);
```

However, a position also holds information about reward tokens owed. As such, if a position that had a reward info with `amount_owed > 0` was burned, the reward tokens would not be redeemable by the position owner.

Recommendation

Recommend including a check to ensure the position does not have any reward tokens to be owed.

Alleviation

[**Turbos Finance**, 05/01/2023]: The team heeded the advice and resolved the issue in commit 21ed8904b65d6e76202b72bcb14cf252d2f82efa by requiring positions to have no reward tokens owed when burning them.

SOR-01 MISSING CORRESPONDING FUNCTIONALITY TO UNLOCK THE POOL

Category	Severity	Location	Status
Logical Issue	Medium	sources/pool.move (04/27-5609659): 554; sources/pool_factory.move (04/27-5609659): 221~227	Resolved

Description

In `pool_factory`, the factory admin has the authority to disable a pool by invoking the `lock_pool` function.

In `pool_factory`:

```
226    pool::lock_pool(pool, ctx);
```

In `pool`:

```
554    pool.unlocked = false;
```

However, there is no corresponding logic to unlock a pool with the `locked` status, which may cause the locked pool to be disabled forever.

Recommendation

Recommend adding corresponding logic to recover the pool from "locked" to avoid locked funds.

Alleviation

[**Turbos Finance**, 05/01/2023]: The team heeded the advice and resolved the issue in commit cc05bcaf8cd3688cbfba31adbb068d92f310b4d5 by allowing locked pools to be unlocked.

I32-01 INCORRECT EUCLIDEAN MODULO FOR NEGATIVE DIVISORS

Category	Severity	Location	Status
Incorrect Calculation	Minor	sources/math/i32.move (04/19-7ffa9e0): 158	Resolved

Description

The Euclidean modulo operation between a dividend `v` and divisor `n` is usually the smallest non-negative number equivalent to `v mod n`. However, the current implementation assumes `n` is positive, allowing the return value to have a higher magnitude than `n`.

```
177     public fun mod_euclidean(v: I32, n: I32): I32 {
178         let r = mod(v, n);
179         if (sign(r) == 1) {
180             add(r, n)
181         } else {
182             r
183         }
184     }
```

Although current use of this function in the project always uses a positive number `n`, future use of this function may cause unexpected outcomes.

Proof of Concept

A unit test is provided to show that the function can return a negative value.

```
#[test]
fun test_euclid_mod() {
    let v = neg_from(1);
    let n = neg_from(2);
    let r = mod_euclidean(v, n);

    assert!(sign(r) == 1, 0);
    assert!(abs_u32(r) == 3, 1);
}
```

Recommendation

Recommend only allowing `n` to be non-negative or handling the case when `n` is negative.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 4351e4b50098876d8db8bdbc8bf198a8bb8f215c by requiring the input `n` to be non-negative.

MOV-01 INCOMPATIBILITY SUI-FRAMEWORK LOCATION FOR NEWER VERSION

Category	Severity	Location	Status
Logical Issue	Minor	Move.toml (04/19-7ffa9e0): 6	Resolved

Description

The location information for `sui-framework` may not be compatible with the new version of the Sui codebase, which may cause failure during building.

The new `sui-framework` location has been updated to `subdir = "crates/sui-framework/packages/sui-framework"`.

Reference: Pull 9618

Recommendation

Recommend update the `sui-framework` location to avoid building failure.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 3af5afa6e3ad84652c8e1fd43fdcde9b8cfe5eb0 by updating Sui's version.

PO3-01 | INCORRECT RETURN VALUE OF `position_id()`

Category	Severity	Location	Status
Inconsistency	Minor	sources/position_nft.move (05/30-374a2aa): 117	Resolved

Description

The function `position_id()` returns the `pool_id` field instead of the `position_id` field.

```
116     public fun position_id(nft: &TurbosPositionNFT): ID {
117         nft.pool_id
118     }
```

Recommendation

It is recommended for the function to return the correct field.

Alleviation

[**Turbos Finance**, 06/12/2023]: The team heeded the advice and resolved the issue in commit 8b7aa572f0136e82c4157bec1fbf565b14d27cd4 by returning the correct value.

PO4-01 | INCORRECT REWARD VALUES USED

Category	Severity	Location	Status
Logical Issue	Minor	sources/pool.move (06/12-reward-fix): 2017	Resolved

Description

For the situation `tick_lower_index < tick_upper_index < tick_current_index`, the Tick objects corresponding to `tick_lower_index` and `tick_upper_index` will have their reward growths set to 0.

```
2015        if (
2016            i32::lt(pool.tick_current_index, tick_lower_index) ||
2017            i32::gt(pool.tick_current_index, tick_upper_index)
2018        ) {
2019            modify_tick_reward_outside(pool, tick_lower_index, 0, 0);
2020            modify_tick_reward_outside(pool, tick_upper_index, 0, 0);
```

However, this case should have the reward growths of the ticks set to the global growth, as shown when updating a tick.

```
            // by convention, we assume that all growth before a tick was
initialized happened _below_ the tick
            if (i32::lte(tick_index, tick_current_index)) {
                tick.fee_growth_outside_a = fee_growth_global_a;
                tick.fee_growth_outside_b = fee_growth_global_b;
                tick.reward_growths_outside = reward_infos;
            };
```

Recommendation

It is recommended to change the reward growths of ticks below the current tick to the global growth.

Alleviation

[**Turbos Finance**, 06/13/2023]: The team heeded the advice and resolved the issue in commit ccf93f266b16df3500ee928dc86e644b4ffe33bc by using the correct values.

PO4-02 | MISSING CASES WHEN MODIFYING TICKS

Category	Severity	Location	Status
Logical Issue	Minor	sources/pool.move (06/12-reward-fix): 2010	Resolved

Description

The function `modify_tick_reward()` currently handles the following cases:

- `current_index < lower_index`
- `current_index > upper_index`
- `lower_index < current_index < upper_index`

However, it is currently not handling the following two cases:

- `lower_index = current_index < upper_index`
- `lower_index < current_index = upper_index`

Recommendation

It is recommended to handle the above missing cases, where ticks less than or equal to the current tick should have their growths set to the global growth and ticks above the current tick should have their growths set to 0.

Alleviation

[**Turbos Finance**, 06/13/2023]: The team heeded the advice and resolved the issue in commit ccf93f266b16df3500ee928dc86e644b4ffe33bc by considering all cases.

POE-01 | MISSING VERSION CHECK ON FLASH SWAP FUNCTIONS

Category	Severity	Location	Status
Inconsistency	Minor	sources/pool.move (Sep-1-flash-and-burn): 470, 511	Resolved

Description

The functions `flash_swap()` and `repay_flash_swap()` do not check the pool's `VERSION` number, unlike other operations. As old packages still exist on-chain in Sui ecosystem, users do not have to use the most updated flash swap functions.

Recommendation

It is recommended to include the shared `Versioned` object as an input and check it against the package's version number.

Alleviation

[**Turbos Finance**, 09/03/2023]: The team heeded the advice and resolved the issue in commit f8c716a17dabc351be6f8018112748d827fdcf14 by adding validation on the flash swap version.

POI-01 | `Burn` SHOULD BE `friend` ONLY

Category	Severity	Location	Status
Logical Issue	Minor	sources/position_nft.move (04/19-7ffa9e0): 57	Resolved

Description

The function `burn()` is used to delete an NFT, but does not contain checks that the associated `Position` object has zero liquidity and zero tokens owed.

```
57      public entry fun burn<CoinTypeA, CoinTypeB, FeeType>(nft: TurbosPositionNFT
<CoinTypeA, CoinTypeB, FeeType>) {
58          let TurbosPositionNFT { id, name: _, description: _, url: _ } = nft;
59          object::delete(id)
60      }
```

This is contrary to the `burn` function in the `position_manager` module, where checks on the associated `Position` object are performed.

```
116     public entry fun burn<CoinTypeA, CoinTypeB, FeeType>(
117         positions: &mut Positions,
118         nft: TurbosPositionNFT<CoinTypeA, CoinTypeB, FeeType>,
119         _ctx: &mut TxContext
120     ) {
121         let nft_address = object::id_address(&nft);
122         let position = dof::borrow_mut<address, Position>(&mut positions.id,
nft_address);
123         assert!(position.liquidity == 0 && position.tokens_owed_a == 0 &&
position.tokens_owed_a == 0, EPositionNotCleared);
124         delete_user_position(positions, nft_address);
125         burn_nft(nft);
126     }
```

As `position_nft::burn()` is a `public entry` function, a user is able to burn their NFT, bypassing the checks in `position_manager::burn()`. This may lead to tokens that are related to the `Position` object to be locked as increasing liquidity, decreasing liquidity, and collecting fees all require the NFT.

Recommendation

Recommend changing the `position_nft::burn()` function to `public(friend)` instead of `public entry`.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 08d38c4837e9b4c432b2b1c0f264f073b67144d4 by changing the function to be friend-only.

POL-01 MISSING INPUT VALIDATION ON TOKEN TYPE WHEN GENERATING THE POOL

Category	Severity	Location	Status
Logical Issue	Minor	sources/pool_factory.move (04/19-7ffa9e0): 114	Resolved

Description

The `pool_factory` module enables users to create new pools using the `deploy_pool` and `deploy_pool_and_mint` functions. Nonetheless, it lacks adequate verification of the provided token type, allowing users to potentially generate a pool with identical tokens on both sides.

Proof of Concept

The following test shows the simulation to create a <BTC, BTC> pool:

```
public fun init_pools2(
        admin: address,
        player: address,
        player2: address,
        scenario: &mut Scenario,
    ) {
        tools_tests::init_fee_type(
            admin,
            scenario
        );

        tools_tests::init_tests_coin(
            admin,
            player,
            player2,
            10000,
            scenario
        );

        tools_tests::init_pool_factory(
            admin,
            scenario
        );


        test_scenario::next_tx(scenario, admin);
        {
            let admin_cap = test_scenario::take_from_sender<PoolFactoryAdminCap>(scenario);
            let pool_config = test_scenario::take_shared<PoolConfig>(scenario);
            let fee_type = test_scenario::take_immutable<Fee<FEE500BPS>>(scenario);

            let sqrt_price = math_sqrt_price::encode_price_sqrt(1, 100);
            pool_factory::deploy_pool<BTC, BTC, FEE500BPS>(
                &mut pool_config,
                &fee_type,
                sqrt_price,
                test_scenario::ctx(scenario),
            );
            test_scenario::return_to_sender(scenario, admin_cap);
            test_scenario::return_shared(pool_config);
            test_scenario::return_immutable(fee_type);
        };
    }

    #[test]
    public fun test_deploy_pool() {
        let admin = @0x0;
        let player = @0x1;
        let player2 = @0x2;

```

```
        let scenario_val = test_scenario::begin(admin);
        let scenario = &mut scenario_val;

        init_pools2(admin, player, player2, scenario);

        test_scenario::end(scenario_val);
    }

```

Recommendation

Recommend adding validation to avoid same token type on both side for new pool pair.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 08d38c4837e9b4c432b2b1c0f264f073b67144d4 by adding a check to ensure different token types are used when creating a pool.

POR-02 | INCORRECT RETURN VALUE IN `next_pool_reward_infos()`

Category	Severity	Location	Status
Logical Issue	Minor	sources/pool.move (05/30-374a2aa): 970	Resolved

Description

The function `next_pool_reward_infos()` returns a vector of zeroes for the case when there is zero liquidity or when the last update time was 0 ms ago.

```
969             if (pool.liquidity == 0 || time_delta == 0) {
970                 vector::insert(&mut growth_global_vector, 0, i);
```

This will cause an underflow issue when updating rewards in `cross_tick()`.

```
1387                let reward_new = vector::borrow(reward_growth_global, i);
1388                let reward = vector::borrow_mut(&mut tick.
reward_growths_outside, i);
1389                *reward = math_u128::wrapping_sub(*reward_new, *reward);
```

The `reward_growth_global` variable is the return value of `next_pool_reward_infos()`, so if `cross_tick()` was invoked when the previous update was 0 ms ago, then `reward_new` would be a vector of zeroes, allowing an underflow situation. An example of this is if someone updated their position and then performed a swap.

Recommendation

It is recommended to return a vector with the current `reward_info.growth_global` values instead of a vector of zeroes in the situation that `pool.liquidity == 0 || time_delta == 0`.

For example,

```
if (pool.liquidity == 0 || time_delta == 0) {
    let reward_info = vector::borrow(&pool.reward_infos, i);
    vector::insert(&mut growth_global_vector, reward_info.growth_global, i);
```

Also, it is recommended to use normal subtraction in situations where underflows should never occur.

Alleviation

[**Turbos Finance**, 06/01/2023]: The team heeded the advice and resolved the issue in commit

51dd68cd443afab6cec8af56f021e3ae0d555113 by revising the `next_pool_reward_info()`.

POU-01 | INCORRECT CONDITION FOR SWAPS

Category	Severity	Location	Status
Logical Issue	Minor	sources/pool.move (04/27-5609659): 404	Resolved

Description

When computing a swap result, a condition is placed so that the inputted `sqrt_price_limit` does not exceed the price bounds.

```
404         if (sqrt_price_limit < MIN_SQRT_PRICE || sqrt_price_limit >
sqrt_price_limit) abort ESqrtPriceOutOfBounds;
```

The price limit is checked to be at least `MIN_SQRT_PRICE`, but the check to be at most `MAX_SQRT_PRICE` is incorrect. The upper bound check is currently `sqrt_price_limit > sqrt_price_limit`, which is never true, allowing users to specify a price limit that is out of bounds.

Recommendation

Recommend replacing `sqrt_price_limit > sqrt_price_limit` with `sqrt_price_limit > MAX_SQRT_PRICE`.

Alleviation

[**Turbos Finance**, 05/01/2023]: The team heeded the advice and resolved the issue in commit bc06ae7005852c1d335e55380a55057e3c851064 by using the correct price condition.

SOU-01 INCORRECT TRANSFER FUNCTIONS ON OBJECTS NOT DEFINED IN MODULES

Category	Severity	Location	Status
Coding Issue	Minor	sources/fee10000bps.move (04/19-7ffa9e0): 19; sources/fee3000bps.move (04/19-7ffa9e0): 19; sources/fee500bps.move (04/19-7ffa9e0): 19; sources/pool.move (04/19-7ffa9e0): 892, 897, 918, 927, 949, 954, 976, 981, 1010, 1015; sources/pool_factory.move (04/19-7ffa9e0): 99, 121; sources/position_manager.move (04/19-7ffa9e0): 337	Resolved

Description

The linked functions in the `transfer` module can only be used on objects defined in the same module. Since "devnet-0.29.0", for objects outside of the module with the `store` keyword, the associated `public_` function needs to be used. For example, `transfer::public_freeze_object()` or `transfer::public_transfer()`.

Reference: f389b8e83e7e408039992a577dfb723d13ba1675

Recommendation

Recommend using the associated `public_` transfer functions on objects defined outside of the module.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 3af5afa6e3ad84652c8e1fd43fdcde9b8cfe5eb0 by using the associated `public_` functions.

SOU-02 | INCONSISTENT FEE LIMIT

Category	Severity	Location	Status
Inconsistency	Minor	sources/math_swap.move (04/19-7ffa9e0): 43; sources/pool.move (04/19-7ffa9e0): 252; sources/pool_factory.move (04/19-7ffa9e0): 131, 142	Resolved

Description

When enabling or changing fees of a pool, the following limit is placed:

```
        assert!(fee < 1000000, EInvalidFee);
```

However, when charging fees, the fee is considered to be out of 10000:

```
 252                 let delta = step_fee_amount * (pool.fee_protocol as u128) /
10000;
 253                 step_fee_amount = step_fee_amount - delta;
```

Due to this inconsistency, it is possible for `delta` to exceed `step_fee_amount`, resulting in an underflow error.

Also note that `math_swap.move` assumes the fee is out of 1000000:

```
41             amount_calc = full_math_u128::mul_div_floor(
42                 amount_remaining,
43                 ((1000000 - fee_rate) as u128),
44                 1000000,
45             );
```

Recommendation

Recommend using a consistent limit on fees.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 3fddddd39f8c7ad2854473c7611bb08666947828 by using a consistent limit of 1000000 for fees.

STR-01 | ZERO CONVERTS TO THE EMPTY STRING

Category	Severity	Location	Status
Logical Issue	Minor	sources/string_tools.move (04/19-7ffa9e0): 80	Resolved

Description

The function `u64_to_string()` transforms a number to a string representation of the number. However, for the number 0, the output string is the empty string. This is because the function ignores all leading zeroes before creation of the string.

```
public fun u64_to_string(number: u64): String {
    let places = 20;
    let base = math::pow(10, 19); //@note highest u64 power of 10
    let i = places;

    let str = &mut string::utf8(vector[]);

    while (i > 0) {
        let quotient = number / base;
        if (quotient != 0) {
            number = number - quotient * base
        };

        if (!string::is_empty(str) || quotient != 0) {
            string::append_utf8(str, vector<u8>[((quotient + 0x30) as u8)])
        };

        base = base / 10;
        i = i - 1;
    };

    *str
}
```

Proof of Concept

A unit test is provided to show that converting the number 0 results in the empty string.

```
#[test]
fun test_zero_string() {
    assert!(u64_to_string(0) == string::utf8(b""), 0);
}
```

Recommendation

Recommend handling the case when `number == 0`.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 6a78b87bd9d58b5aba09d10901eef905acc0a17a by separately handling the case of converting the number 0.

SWA-01 | MISSING VALIDATION ON `_deadline`

Category	Severity	Location	Status
Logical Issue	Minor	sources/swap_router.move (04/19-7ffa9e0): 23, 54	Resolved

Description

In `swap_router.move`, the `_deadline` variable is announced in swap functions but not used to avoid expired swaps.

Recommendation

Recommend adding validation on `_deadline` to avoid unexpected expired swap operations.

Reference:

1. Sui devnet-0.29.0
2. Accessing Time in Sui Move

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit c6fa2e82332b1ac207679b818a90e8ddeb327f5d by adding timestamp checks.

SWA-02 | UNUSED VARIABLE AND POTENTIAL MISSING VALIDATION

Category	Severity	Location	Status
Logical Issue	Minor	sources/swap_router.move (04/19-7ffa9e0): 19, 50, 82	Resolved

Description

In the linked functions, the input variable `_amount_out_min` is not used to check that the swap result fulfilled the minimum swap result expectation:

- `swap_a_b()`
- `swap_b_a()`
- `swap_a_b_c()`

Recommendation

Recommend adding validation on `_amount_out_min` in aforementioned methods to avoid unexpected swap results.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit af11f5646afe48ef5aee24e3765e94fdf6c1c87b by adding checks on thresholds.

GLOBAL-01 | MISSING EMIT EVENTS

Category	Severity	Location	Status
Coding Style	Informational		Resolved

Description

In the current codebase, there are no events being triggered when there is a update to the state, such as changes in liquidity or pool updates. By emitting events during these state changes, users would be able to stay informed about updates to the contract.

Reference: Sui events

Recommendation

We recommend adding events for state-changing actions, and emitting them in their relevant functions.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 27082242561705ae765e549d57c415262934868b by events for several important functions.

GLOBAL-02 | TICK MODIFICATION

Category	Severity	Location	Status
Inconsistency	Informational		Resolved

Description

In commit 51dd68cd443afab6cec8af56f021e3ae0d555113, a function was added to correct reward growths for certain Ticks. However, this may not be enough if a Position object had their reward growth and amount owed updated using the problematic Tick.

There should also be functionality to change the `pool::Position.reward_infos` and `position_manager::Position.reward_infos` fields for affected positions.

The `reward_infos.reward_growth_inside` should have a maximum value of `Tick.reward_growths_outside` while the `reward_infos.amount_owed` value should be sets to a reasonable value.

Recommendation

It is recommended to add a function to change a position's rewards.

Alleviation

[**Turbos Finance**, 06/13/2023]: The team heeded the advice and resolved the issue in commit 9ec7cfa9fe40d6a9a225b664554c8e2b26825901 by adding functionality to change a position's rewards.

PO4-03 | INVALID ASSIGNMENT ISSUE

Category	Severity	Location	Status
Coding Style	Informational	sources/pool.move (06/12-reward-fix): 1447	Resolved

Description

In commit 4b670b2f30aed612881668afb2b5df4d3e90f6b1, which tries to fix the discussed concern, an invalid assignment lies on line 1447 that `fee_growth_below_b = pool.fee_growth_global_a;`.

Although we believe that this branch will not get involved in the business logic because `update_tick` is always called before the `next_fee_growth_inside()`. A tick can only have `initialized == false` if it was just created or cleared, which means `liquidity_gross == 0`, which will cause the tick to have `initialized == true` in `update_tick()`.

Recommendation

Recommend revising the aforementioned assignment.

Alleviation

[**Turbos Finance**, 06/13/2023]: The team heeded the advice and resolved the issue in commit 6f41da280d0aba8647a37892329cfb3d85ac39ad by using the correct assignment.

POF-02 | TEST ONLY FUNCTION

Category	Severity	Location	Status
Coding Issue	Informational	sources/pool_factory.move (04/27-5609659): 137	Resolved

Description

The `coin_to_vec` function is solely utilized in the test functions and not needed for the project's operations..

Recommendation

If this function is intended to be a test function, it should be decorated with `#[test_only]`.

Alleviation

[**Turbos Finance**, 05/01/2023]: The team heeded the advice and resolved the issue in commit 4f67b551dc02300010b5ea77edecc65b8ce31722 by removing the function.

POL-02 | POSSIBLE TO DEPLOY DUPLICATE POOLS

Category	Severity	Location	Status
Logical Issue	Informational	sources/pool_factory.move (04/19-7ffa9e0): 54~55	Resolved

Description

The current codebase allows users to generate pools with the same token pair and the same fee setting. For example, it is possible to create two BTC/USDC pools, both with the 500BPS fee setting.

This is not a good economically as the liquidity for a token pair may be diluted into multiple pools.

Proof of Concept

The following test will simulate creating two identical pools and printing out the pool UID:

```
    public fun init_pools(
        admin: address,
        player: address,
        player2: address,
        scenario: &mut Scenario,
    ) {
        tools_tests::init_fee_type(
            admin,
            scenario
        );

        tools_tests::init_tests_coin(
            admin,
            player,
            player2,
            10000,
            scenario
        );

        tools_tests::init_pool_factory(
            admin,
            scenario
        );

        // FIRST POOL
        test_scenario::next_tx(scenario, admin);
        {
            let admin_cap = test_scenario::take_from_sender<PoolFactoryAdminCap>(scenario);
            let pool_config = test_scenario::take_shared<PoolConfig>(scenario);
            let fee_type = test_scenario::take_immutable<Fee<FEE500BPS>>(scenario);

            let sqrt_price = math_sqrt_price::encode_price_sqrt(1, 100);
            pool_factory::deploy_pool<BTC, USDC, FEE500BPS>(
                &mut pool_config,
                &fee_type,
                sqrt_price,
                test_scenario::ctx(scenario),
            );
            test_scenario::return_to_sender(scenario, admin_cap);
            test_scenario::return_shared(pool_config);
            test_scenario::return_immutable(fee_type);
        };


        // SECOND POOL
        test_scenario::next_tx(scenario, admin);
        {
            let admin_cap = test_scenario::take_from_sender<PoolFactoryAdminCap>(scenario);
            let pool_config = test_scenario::take_shared<PoolConfig>(scenario);
```

```
            let fee_type = test_scenario::take_immutable<Fee<FEE500BPS>>(scenario);

            let sqrt_price = math_sqrt_price::encode_price_sqrt(1, 100);
            pool_factory::deploy_pool<BTC, USDC, FEE500BPS>(
                &mut pool_config,
                &fee_type,
                sqrt_price,
                test_scenario::ctx(scenario),
            );
            test_scenario::return_to_sender(scenario, admin_cap);
            test_scenario::return_shared(pool_config);
            test_scenario::return_immutable(fee_type);
        };
    }

    #[test]
    public fun test_deploy_pool() {
        let admin = @0x0;
        let player = @0x1;
        let player2 = @0x2;

        let scenario_val = test_scenario::begin(admin);
        let scenario = &mut scenario_val;

        init_pools(admin, player, player2, scenario);

        test_scenario::end(scenario_val);
    }
```

Output:

```
[debug] 0x2::object::ID {
  bytes: @0x751dd9619c0a0e38d58e91544a662ec57362fce1a82a20db2fd1e77b86dc5a45
}
[debug] 0x2::object::ID {
  bytes: @0x9afbd30b749add9d3214c18239f9453011305c0648ad9388d11545b9e5a7ef32
}
```

Recommendation

If unintended, it is recommended to have checks so that different pools have different tokens or fee settings.

Alleviation

[**Turbos Finance**, 05/03/2023]: The team heeded the advice and resolved the issue in commits 61954af9a66c8e5c7dd3c5abc3a5dab51bdc9fb7 and 9463fce767877b3848a7d9e4dbd7f1c6e17c2dc3 by checking to ensure that pools cannot use the same token pair and fee type.

POT-02 | REWARD DISTRIBUTION

Category	Severity	Location	Status
Logical Issue	Informational	sources/position_manager.move (04/27-5609659): 401	Partially Resolved

Description

In the project, the `reward_manager` is authorized to initialize the reward plan to the pool by providing the reward token. However, it should be noted that the reward payment is not guaranteed if the `reward_manager` fails to offer sufficient tokens to the reward vault.

Recommendation

Recommend having mechanisms to ensure there are sufficient rewards for users.

Alleviation

[**Turbos Finance**, 05/01/2023]: The team heeded the advice and partially resolved the issue in commit b709785dfda20aca427ce72261deecfb9a4ee65c by adding a function that allows replacement of the reward manager.

The team will review the project's reward plan, and in extreme cases, revoke the manager's permissions and reset emission to avoid this issue.

[**CertiK**, 05/01/2023]: This issue is partially resolved as the changes do not guarantee rewards, but do help deal with the issue.

SWA-03 | MISSING VALIDATION ON EMPTY VECTOR

Category	Severity	Location	Status
Logical Issue	Informational	sources/swap_router.move (04/19-7ffa9e0): 17, 48, 80	Resolved

Description

The linked vector variables are missing length validation to avoid empty vectors. Since all linked variables will be used as input for `pool::merge_coin` and will be processed with `vector::pop_back` later, which will raise error when the vector is empty:

```
public(friend) fun merge_coin<CoinType>(
    coins: vector<Coin<CoinType>>,
): Coin<CoinType> {
    let self = vector::pop_back(&mut coins);
    ...
}
```

Recommendation

Recommend adding length validation on the linked variables to stop an empty vector input at an earlier stage and save computational cost.

Alleviation

[**Turbos Finance**, 04/27/2023]: The team heeded the advice and resolved the issue in commit 92edf001256586c65870c1aeb99faabe91c3e00f by adding validation for empty inputs.

APPENDIX | TURBOS FINANCE - AUDIT

Finding Categories

Categories	Description
Coding Style	Coding Style findings may not affect code behavior, but indicate areas where coding practices can be improved to make the code more understandable and maintainable.
Coding Issue	Coding Issue findings are about general code quality including, but not limited to, coding mistakes, compile errors, and performance issues.
Incorrect Calculation	Incorrect Calculation findings are about issues in numeric computation such as rounding errors, overflows, out-of-bounds and any computation that is not intended.
Inconsistency	Inconsistency findings refer to different parts of code that are not consistent or code that does not behave according to its specification.
Logical Issue	Logical Issue findings indicate general implementation issues related to the program logic.
Centralization	Centralization findings detail the design choices of designating privileged roles or other centralized controls over the code.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

DISCLAIMER | CERTIK

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR

UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

CertiK | Securing the Web3 World

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.