CERTIK

# Security Assessment

# Francium Protocol

Nov 16th, 2021

# Table of Contents

## Summary

This report has been prepared for Francium Protocol to locate potential vulnerabilities and thereafter verify the correctness of specific components in said project's source code. A series of thorough security assessments have been performed utilizing the Manual Review technique, the goal of which is to help the client protect their users through discovering, mitigating and ultimately fixing security flaws that could lead to unauthorized access, loss of funds, cascading failures, and/or other vulnerabilities. Alongside each security finding a recommendation on fixes and/or mitigation methods are also given.

# Overview

## Project Summary

| | |
|---|---|
| Project Name | Francium Protocol |
| Platform | Solana |
| Language | Rust |
| Codebase | https://github.com/Francium-DeFi/francium-strategy-contracts/tree/main/lyf-raydium |
| Commit | 4edf082be93015777e9f848544c02daa7b6684e3 |

## Audit Summary

| | |
|---|---|
| Delivery Date | Nov 16, 2021 |
| Audit Methodology | Manual Review |
| Key Components | |

## Vulnerability Summary

| Vulnerability Level | Total | Pending | Declined | Acknowledged | Partially Resolved | Resolved |
|---|---|---|---|---|---|---|
| ● Critical | 0 | 0 | 0 | 0 | 0 | 0 |
| ● Major | 1 | 0 | 0 | 0 | 0 | 1 |
| ● Medium | 2 | 0 | 0 | 0 | 0 | 2 |
| ● Minor | 10 | 0 | 0 | 2 | 0 | 8 |
| ● Informational | 8 | 0 | 0 | 4 | 0 | 4 |
| ● Discussion | 0 | 0 | 0 | 0 | 0 | 0 |

## Audit Scope

| ID | Commit | File | SHA256 Checksum |
|---|---|---|---|
| LEN | 4edf082 | lyf-raydium/programs/lyf-raydium/src/adapter/lending_pool.rs | 8daf71ef063a68ffe5b37e2d2496f85a97a4b3521a6e16d68c872db01dd1232d |
| MOD | 4edf082 | lyf-raydium/programs/lyf-raydium/src/adapter/mod.rs | 74989c62eb7c3eb303f5f05e83f7475fb91eb7d3738b25a3577e451240e6e651 |
| RAY | 4edf082 | lyf-raydium/programs/lyf-raydium/src/adapter/raydium.rs | bd68b330255c11adb675a15acdb171b146dadf10054f625a84b54ae9e05794f5 |
| LIB | 4edf082 | lyf-raydium/programs/lyf-raydium/src/lib.rs | 1e398c73898362665a026b5d5231004f8aae812dedd421a932d90198001342f9 |
| TYP | 4edf082 | lyf-raydium/programs/lyf-raydium/src/types.rs | e606c284279b59eae004e300cdfcfc646d2f3ae2b7c982520c59d33d41f03653 |
| CKP | 4edf082 | lyf-raydium/programs/lyf-raydium/src/version.rs | 436609231682191818 27a1e1cf4f2715a9d49ebbc4c205a1683dcd02245ad096 |
| CCP | 4edf082 | lyf-raydium/programs/lyf-raydium/Cargo.toml | 49ee9bcdea03f8ce268036ad3ab9f989454c1230eb46c4189eb8a2656c56b4b7 |
| ACK | 4edf082 | lyf-raydium/Anchor.toml | ba0f5421881cb6473e2c5bb694e3d0e85ad9b12140d9ec61502b58d3dbeb0f5e |
| CCK | 4edf082 | lyf-raydium/Cargo.toml | 4ec7725ef223b05c64e33af9d3c7ad116e9376f30e6a9830a9d1e0ec13b051b8 |

# Review Notes

Our audit approach primarily revolves around a multi-round manual review process, and largely favors modularity and encapsulation in code design. At a high level we analyze each object (or module) by their interfaces and references to other objects. This ultimately ensures that the same security properties can be extended to new objects added to the system, which in return minimizes the attack surface of the application down to the implementation of specific objects.

Additionally we analyze how the state machines are defined and how state transitions are triggered, the focus of which is to check the implementation against the specs (if provided) and hence mitigate the possibilities of unintentional state behaviors taking place.

## Key Checks

### Common Vulnerabilities

- Constants precision and conversion
- Integer overflow/underflow
- Stack overflow
- Index Out-of-Bound
- Out-of-Memory

### Ownership

- Moves (e.g. control flow, indexed content)
- Shared ownership (e.g. reference-counted pointer types)

### References

- Sharing and mutation
- Borrowing references
- Receiving references as parameters
- Returning references

### Composition

- Type grouping
- Cascading changes

### Decoupling

- Semantic consistency
- Indirection and allocation cost
- Type coercion
- Trait pollution

## Error Handling

- Unwrapping, logging and propagating errors
- Panics (e.g. detection, unwinding and recovery)

## Unsafe Code

- Undefined behaviors (e.g. memory leaks, use after fee, double free)
- Exception safety
- Uninitialized memory
- Data races

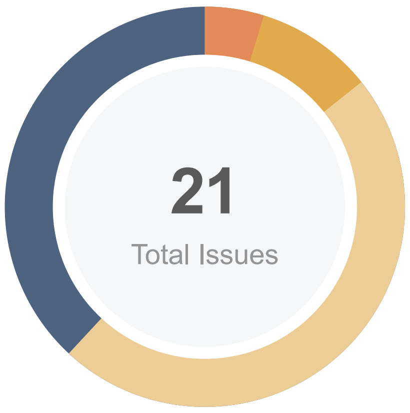
## Advanced Vulnerabilities

- Format string attacks
- Cryptographic attacks (e.g. timing attacks)

## General Checks

- Organization of crates and modules
- Language best practices

# Findings



| | | |
|---|---|---|
| 🟥 | **Critical** | **0** (0.00%) |
| 🟧 | **Major** | **1** (4.76%) |
| 🟨 | **Medium** | **2** (9.52%) |
| 🟨 | **Minor** | **10** (47.62%) |
| 🟦 | **Informational** | **8** (38.10%) |
| 🟩 | **Discussion** | **0** (0.00%) |

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| FRA-01 | Insufficient & Inconsistent Error Handling | Control Flow | ● Minor | ✓ Resolved |
| CKP-01 | Missing User Credential Checks | Inconsistency, Coding Style | ● Medium | ✓ Resolved |
| LIB-01 | Unused Imports and Constants | Coding Style | ● Informational | ⓘ Acknowledged |
| LIB-02 | Inconsistency in Variable Naming | Coding Style, Inconsistency | ● Informational | ⓘ Acknowledged |
| LIB-03 | Redundant Function Argument | Inconsistency, Volatile Code | ● Minor | ⓘ Acknowledged |
| LIB-04 | Insufficient Check for Rewards Transfer | Logical Issue | ● Minor | ✓ Resolved |
| LIB-05 | Missing Amount Validation | Logical Issue | ● Minor | ✓ Resolved |
| LIB-06 | Hardcoded Slippage Approximation | Inconsistency | ● Minor | ⓘ Acknowledged |
| LIB-07 | Insufficient Conditional Check | Logical Issue | ● Informational | ✓ Resolved |
| LIB-08 | Missing Check For Insufficient Liquidity | Logical Issue | ● Minor | ✓ Resolved |
| LIB-09 | Missing Check For Last Updated Slot | Logical Issue, Inconsistency | ● Minor | ✓ Resolved |
| LIB-10 | Missing Pending Withdraw Flag Check | Logical Issue, Inconsistency | ● Medium | ✓ Resolved |
| LIB-11 | Liquidity Addition Not Allowed For a Single Token | Logical Issue | ● Minor | ✓ Resolved |

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| LIB-12 | Hardcoded Initialization Values | Magic Numbers | ● Informational | ⓘ Acknowledged |
| RAY-01 | Redundant Closure | Coding Style, Language Specific | ● Informational | ✓ Resolved |
| RAY-02 | Inconsistent Comparison | Inconsistency | ● Informational | ✓ Resolved |
| RAY-03 | Inconsistent Comments and Code | Logical Issue, Inconsistency | ● Minor | ✓ Resolved |
| RAY-04 | Logical Inconsistency in Function Implementation | Control Flow | ● Major | ✓ Resolved |
| RAY-05 | Hardcoded Value | Coding Style | ● Informational | ⓘ Acknowledged |
| RAY-06 | Insufficient Access Control | Logical Issue, Inconsistency | ● Minor | ✓ Resolved |
| RAY-07 | Typo in Comment | Coding Style | ● Informational | ✓ Resolved |

## FRA-01 | Insufficient & Inconsistent Error Handling

| Category | Severity | Location | Status |
|---|---|---|---|
| Control Flow | ● Minor | Global | ✓ Resolved |

### Description

There is generally a lack of consistent error handling and propagation in the codebase:

- Math overflow errors are called upon with `unwrap()` which can cause panics
- Associated functions like `update_platform_rewards` and `unstake_lp` in `lib.rs` return the execution midway once a certain condition is met, without returning possible errors.

### Recommendation

Add proper error handling to all occurrences. For example:

- Return a `Result<T, ProgramError>` from all the mathematical operations (e.g. `add_tkn_0` , `sub_tkn_1` etc) so that the methods calling them can handle the negative path accordingly using pattern matching
- Replace `unwrap()` in mathematical operations with `?` to propagate errors up the call stack and avoid panics at runtime
- Return `update_platform_rewards` and `unstake_lp` with proper error messages in case of failure

### Alleviation

The Francium team replied with the following remark:

"If an error occurs during the execution paths, the transaction will fail and the security of the contract will not be affected."

We agree with the team's sentiment and consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## CKP-01 | Missing User Credential Checks

| Category | Severity | Location | Status |
|---|---|---|---|
| Inconsistency, Coding Style | ● Medium | projects/francium-strategy-contracts-main/lending-pool/src/lib.rs (4edf082): 2193~2658 | ✓ Resolved |

### Description

Methods implemented for `StrategyState` lack proper user credential checks before proceeding down the execution path.

### Recommendation

Consider adding a call to function `associated_user_info_account()` since the public keys required by said function can be retrieved from `StrategyState` as well as `UserInfo`. Doing this would allow external functions to access the credential check within `StrategyState` to perform user credential checks.

### Alleviation

The Francium team replied with the following comments:

"Currently the UserInfo check is done when the transaction accounts are resolved or in the process function, so although there is noverification done here (in strategyState), it does not affect the security."

We agree with the team's sentiment and consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## LIB-01 | Unused Imports and Constants

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Style | ● Informational | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 22, 46, 32~33, 33, 34, 36, 374 | ⓘ Acknowledged |

### Description

The following imports aren't being used:

- Line 22 in `lyf_raydium/src/lib.rs`
- Line 46 in `lyf_raydium/src/lib.rs`
- `MIN_LEVERAGE`, `DEFAULT_LEVERAG`E, `LIQUIDATE_LINE_DEFAULT` in `lyf_raydium/src/lib.rs`
- Line 374 in `lyf_raydium/src/lib.rs`

### Recommendation

Remove unused declarations and imports.

### Alleviation

The Francium team acknowledged the issue and decided not to provide an immediate fix considering the code has been deployed and is currently run in production. We consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## LIB-02 | Inconsistency in Variable Naming

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Style, Inconsistency | ● Informational | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 406, 576 | ⓘ Acknowledged |

### Description

Variable `swap_fee_numerator`, `swap_fee_denominator` in the swap function are not named using the underscore prefix to indicate a potentially unused variable. The same applies to function `add_liquidity` declared on line 534.

### Recommendation

Add _ as a prefix for both `swap_fee_numerator` and `swap_fee_denominator` in the swap function and the `add_liquidity` function. Make sure variable naming convention is consistent throughout the codebase, either no _ prefixes are used anywhere when the same information is stored, or used consistently in all places.

### Alleviation

The Francium team acknowledged the issue and decided not to provide an immediate fix considering the code has been deployed and is currently run in production. We consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## LIB-03 | Redundant Function Argument

| Category | Severity | Location | Status |
|---|---|---|---|
| Inconsistency, Volatile Code | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 374 | ⓘ Acknowledged |

### Description

Parameter `param` in the swap function is not used in the function body.

### Recommendation

Remove the argument.

### Alleviation

The Francium team acknowledged the issue and decided not to provide an immediate fix considering the code has been deployed and is currently run in production. We consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## LIB-04 | Insufficient Check for Rewards Transfer

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 116~118 | ✓ Resolved |

### Description

The current implementation would prohibit transferring rewards from the admin account to the strategy account even when `rewards_start_slot` and `rewards_end_slot` happen to be equal.

### Recommendation

Change to `rewards_start_slot >= rewards_end_slot` to avoid unnecessary transfers.

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

## LIB-05 | Missing Amount Validation

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 857~867 | ✓ Resolved |

### Description

There is no check to validate the `amount_in` value.

### Recommendation

Return an error with appropriate error handling if `amount_in < 1`.

### Alleviation

The Francium team reponded with the following comment:

"`amount_in` is the amount of swap needed, and less than 1 means no swap is needed. Therefore there is no need to process the swap and there is no security concern."

We agree with the team's sentiment and consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## LIB-06 | Hardcoded Slippage Approximation

| Category | Severity | Location | Status |
|---|---|---|---|
| Inconsistency | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 466~467, 1304~1305, 507, 1331 | ⓘ Acknowledged |

### Description

Slippage approximation is hardcoded in the call to raydium token swap.

### Recommendation

Implement a consistent and unified slippage approximation and perform calculations accordingly, along with proper overflow checks and error handling.

### Alleviation

The Francium team acknowledged the issue and decided not to provide an immediate fix considering the code has been deployed and is currently run in production. We consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## LIB-07 | Insufficient Conditional Check

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | ● Informational | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 1271~1274 | ✓ Resolved |

### Description

The current implementation, `amount_in <= 1`, would reject swaps when `amount_in` is exactly 1.

### Recommendation

Consider changing the conditional check to `amount_in < 1` so that when `amount_in` is exactly 1 swaps are still permitted.

### Alleviation

The Francium team responded with the following statement:

"When the amount is 1 usually it means the amount is very small (less than 10^-6), which is too small for a swap."

We agree with the team's sentiment and consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## LIB-08 | Missing Check For Insufficient Liquidity

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 374~375 | ✓ Resolved |

### Description

There's currently no checks for insufficient liquidity before making a swap.

### Recommendation

Check the reserves first whether there is a sufficient liquidity for the swap, then proceed with the swap.

### Alleviation

The Francium team responded with the following statement:

"If the balance is insufficient, swap will fail and cause the transaction to fail. Therefore, there is no need to additionally check for insufficient balance."

We agree with the team's sentiment and consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## LIB-09 | Missing Check For Last Updated Slot

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue, Inconsistency | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 2753~2758, 2393~2398 | Resolved |

### Description

There is currently no checks to verify the validity of the new `last_update_slot` value before updating it with the `current_slot` value.

### Recommendation

Check if `self.last_update_slot < current_slot` before `last_update_slot` is updated. Additionally, return an error with proper handling in case of failure.

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

## LIB-10 | Missing Pending Withdraw Flag Check

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue, Inconsistency | ● Medium | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 1080, 1162 | ✓ Resolved |

### Description

The highlighted functions do not have a check for unfinished `pending_withdraw_flag`.

### Recommendation

Similar to `transfer`, `borrow`, `swap`, and `add_liquidity`, add a check for unfinished pending withdrawals.

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

## LIB-11 | Liquidity Addition Not Allowed For a Single Token

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 553~555 | ✓ Resolved |

### Description

The current check, `tkn_0 <= 1 || tkn_1 <= 1`, prohibits liquidity addition when `tkn_0` and/or `tkn_1` are exactly 1.

### Recommendation

Consider prohibiting liquidity addition only when `tkn_0 < 1 || tkn_1 < 1` or explain in comments why it should be otherwise.

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

## LIB-12 | Hardcoded Initialization Values

| Category | Severity | Location | Status |
|---|---|---|---|
| Magic Numbers | ● Informational | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/lib.rs (4edf082): 2251~2253 | ⓘ Acknowledged |

### Description

The initial parameters for `StrategyState` have hardcoded values in them.

### Recommendation

Pull the hardcoded values out and declare them as constants.

### Alleviation

The Francium team acknowledged the issue and decided not to provide an immediate fix. We consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## RAY-01 | Redundant Closure

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Style, Language Specific | ● Informational | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/adapter/raydium.rs (4edf082): 247 | ✓ Resolved |

### Description

The closure in `open_orders = RefMut::map(data, |data| from_bytes_mut(data));` serves no apparent purposes and can be replaced by `from_bytes_mut`.

### Recommendation

Replace `open_orders = RefMut::map(data, |data| from_bytes_mut(data));` with `open_orders = RefMut::map(data, from_bytes_mut);`

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

## RAY-02 | Inconsistent Comparison

| Category | Severity | Location | Status |
|---|---|---|---|
| Inconsistency | ● Informational | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/adapter/raydium.rs (4edf082): 432 | ✓ Resolved |

### Description

The comparison for the user's PC amount and the user's coin amount is inconsistent.

### Recommendation

Either change the second expression to a GTE (>=) or change the first to GT (>).

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

## RAY-03 | Inconsistent Comments and Code

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue, Inconsistency | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/adapter/raydium.rs (4edf082): 380 | ✓ Resolved |

### Description

The comment calls for two conditional checks on the user's balance yet there is no such checks implemented in the code.

### Recommendation

Add appropriate checks before returning to ensure the user account maintains a sufficient balance.

### Alleviation

As an alternative fix the comment has been deleted.

## RAY-04 | Logical Inconsistency in Function Implementation

| Category | Severity | Location | Status |
|---|---|---|---|
| Control Flow | ● Major | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/adapter/raydium.rs (4edf082): 381~474 | ✓ Resolved |

### Description

The current implementation doesn't seem to consider the case when the user's account doesn't have enough balance of either of the two tokens. Function `quote_swap_amount_for_withdraw()` takes an argument to determine the type of token to be withdrawn and after checking if the user has sufficient balance of that token, returns the swap direction and amount. It checks if the user has enough balance for the other token, and if so enables a swap between the two, so that the user can deposit the other token, for a withdrawal of the token dictated by `withdraw_type`. However, if the user doesn't have enough tokens of the other type, then they have to first swap to increase the balance of that token.

### Recommendation

Add a check to see if the user's account has enough tokens on either end to make the withdrawal.

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

## RAY-05 | Hardcoded Value

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Style | ● Informational | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/adapter/raydium.rs (4edf082): 206, 215 | ⓘ Acknowledged |

### Description

The value `1_000_000u128` can be moved and declared as a constant outside the highlighted functions to better demonstrate what it's meant for.

### Recommendation

Declare a constant for `1_000_000u128` and refer to it in the functions.

### Alleviation

The Francium team acknowledged the issue and decided not to provide an immediate fix considering the code has been deployed and is currently run in production. We consider the exhibit fully attended to as it doesn't pose immediate security concerns.

## RAY-06 | Insufficient Access Control

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue, Inconsistency | ● Minor | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/adapter/raydium.rs (4edf082): 594 | ✓ Resolved |

### Description

The highlighted function permits writing into the `amm_id`'s account when it's not supposed to. Allowing this would give `raydium_add_liquidity_v4()` the permission to write into `amm_authority` using `AccountMeta::new`, which would trickle down to `raydium_token_swap()`, `raydium_remove_liquidity_v4()` and other functions that operate on the AMM's balances.

### Recommendation

Replace with `AccountMeta::new_readonly(*amm_authority.key, false),`

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

## RAY-07 | Typo in Comment

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Style | ● Informational | projects/francium-strategy-contracts-main/lyf-raydium/programs/lyf-raydium/src/adapter/raydium.rs (4edf082): 648 | ✓ Resolved |

### Description

The comment is not consistent with the function name.

### Recommendation

Change the comment to :

```
// raydium_remove_liquidity_v4
```

### Alleviation

A fix has been applied at commit 2083b3aefba0bc6a849432b1c9137530bb6d3480.

# Appendix

## 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

"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.

## About

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.