Sienna

Vesting 2.0

Security Assessment

April 28th, 2021

Disclaimer

CertiK reports are not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. These reports are 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 review.

CertiK Reports do 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.

CertiK Reports should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

CertiK Reports represent an extensive auditing 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.

What is a CertiK report?

- A document describing in detail an in depth analysis of a particular piece(s) of source code provided to CertiK by a Client.
- An organized collection of testing results, analysis and inferences made about the structure, implementation and overall best practices of a particular piece of source code.
- Representation that a Client of CertiK has completed a round of auditing with the intention to increase the quality of the company/product's IT infrastructure and or source code.

Overview

Project Summary

Project Name	Sienna - Vesting 2.0
Description	Sienna is a privacy-first, decentralized financing platform.
Platform	Rust
Codebase	GitHub Repository
Commits	1. fcf5822da9ca9c839358f485fd0611535c1a5a24 2. 32f7c75b6545785896832b5d4c1bc7d5ea45ba98

Audit Summary

Delivery Date	April 28th, 2021
Method of Audit	Static Analysis, Manual Review
Consultants Engaged	2
Timeline	April 11th, 2021 - April 28th, 2021

Vulnerability Summary

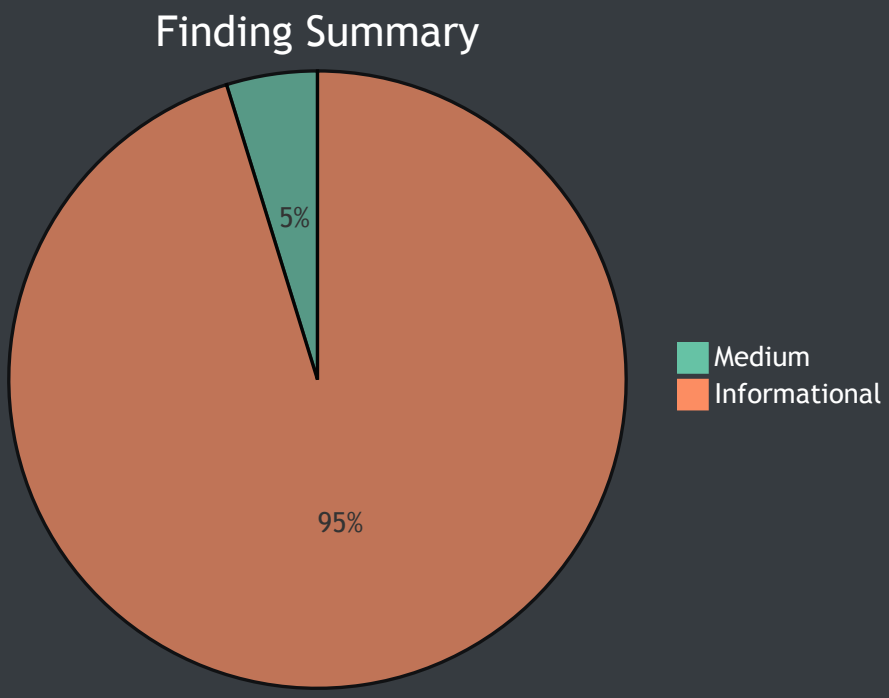
Total Issues	21
Total Critical	0
Total Major	0
Total Medium	1
Total Minor	0
Total Informational	20
Total Resolved	18
Total Acknoledged	3

Executive Summary

Certik was assigned to audit the codebase of the Sienna Vesting mechanism based on Cosmos SDK. The audit has identified issues that ranged from informational to medium and fixed by the team in the alleviation phase.

Files In Scope

ID	Contract	Location
CON	lib.rs	contracts/mgmt/src/lib.rs
CON	lib.rs	contracts/rpt/src/lib.rs
LIB	lib.rs	libraries/linear-map/lib/lib.rs
CAN	canon.rs	libraries/schedule/lib/canon.rs
ERR	errors.rs	libraries/schedule/lib/errors.rs
LIB	lib.rs	libraries/schedule/lib/lib.rs
MUT	mutate.rs	libraries/schedule/lib/mutate.rs
VAL	validate.rs	libraries/schedule/lib/validate.rs
VES	vesting.rs	libraries/schedule/lib/vesting.rs
LIB	lib.rs	libraries/utils/src/lib.rs
STO	storage.rs	libraries/utils/src/storage.rs
VIE	viewing_key.rs	libraries/utils/src/viewing_key.rs


Finding Summary
5%
95%
Medium
Informational

Manual Review Findings

ID	Title	Type	Severity	Resolved
CON-01	Inefficient Error Design	Coding Style	Informational	✓
CON-02	Redundant Variable	Logical Issue	Informational	⟳
CON-03	Code Design Issue	Language Specific	Informational	✓
CON-04	Code Design Issue	Coding Style	Informational	✓
CON-05	Variable Naming	Coding Style	Informational	✓
CON-06	Inefficient Error Design	Coding Style	Informational	✓
CON-07	No Use Of Abstraction	Coding Style	Informational	⟳
CON-08	Redundant Cloning	Language Specific	Informational	✓
LIB-01	If Else Instead Of Match	Language Specific	Informational	⟳
ERR-01	Inefficient Parameter	Logical Issue	Informational	✓
LIB-02	Unclear Implementation Naming	Logical Issue	Informational	✓
LIB-03	Code Structure	Logical Issue	Informational	✓
LIB-04	Code Structure	Logical Issue	Informational	✓
LIB-05	Unclear Implementation Naming	Logical Issue	Informational	✓
MUT-01	Inefficient Parameter	Logical Issue	Informational	✓
MUT-02	Unnecesary Return	Logical Issue	Informational	✓
VAL-01	Implement Validation	Logical Issue	Informational	✓

	Trait			
VES-01	Error Handling Implementation Missing	Logical Issue	Medium	✓
VES-02	Implement Unlock Trait	Logical Issue	Informational	✓
VES-03	Simplification On Match Pattern	Logical Issue	Informational	✓
VIE-01	Hard Coded Value	Language Specific	Informational	✓

CON-01: Inefficient Error Design

Type	Severity	Location
Coding Style	Informational	lib.rs L25-L31

Description:

The error design contains names and string representations that could be more specific to the issue.

Recommendation:

Consider renaming the errors to represent more accurate the issue.

Alleviation:

The team has fixed the issue in commit 8c5a1f7cfbf7df17d61f8026b0731fcc6769a259

CON-02: Redundant Variable

Type	Severity	Location
Logical Issue	Informational	lib.rs L78

Description:

The code creates a new variable that is redundant.

Recommendation:

Consider removing the variable and use the function parameter directly.

Alleviation:

The team opted to not alleviate the issue in this iteration.

CON-03: Code Design Issue

Type	Severity	Location
Language Specific	Informational	lib.rs L172

Description:

The code performs checks and returns in a non optimal way.

Recommendation:

Consider using a match pattern here and removing the redundant return.

Alleviation:

The team fixed the issue in commit up to 0a52dd41b081953702585f9e2cdc6e4033f6a15e

CON-04: Code Design Issue

Type	Severity	Location
Coding Style	Informational	lib.rs L176-L194

Description:

The code contains two very similiar functions.

Recommendation:

Consider refactoring the code and perfom the functionality in a single function.

Alleviation:

The team opted to not alleviate the issue in this iteration.

CON-05: Variable Naming

Type	Severity	Location
Coding Style	Informational	lib.rs L220

Description:

The code creates a new variable with the same name although it has another form.

Recommendation:

Consider renaming the variable to humman_addr.

Alleviation:

The team has fixed the issue in commit 129d7a59fc0112a67615f9bc11670c21b6a3b631

CON-06: Inefficient Error Design

Type	Severity	Location
Coding Style	Informational	lib.rs L39, L41

Description:

The error design contains names and string representations that could be more specific to the issue.

Recommendation:

Consider renaming the errors to represent more accurate the issue.

Alleviation:

The team has fixed the issue in commit c002c3c8792fd7139c93120c7a9469190e4d0318.

CON-07: No Use Of Abstraction

Type	Severity	Location
Coding Style	Informational	lib.rs L155

Description:

The code does not use the language abstractions to calculate the sum.

Recommendation:

Consider using the sum function.

Alleviation:

The team opted to not alleviate the issue in this iteration.

CON-08: Redundant Cloning

Type	Severity	Location
Language Specific	Informational	lib.rs L172

Description:

The code performs a redundant cloning of the variable.

Recommendation:

Consider removing the cloning.

Alleviation:

The team has fixed the issue in commit 988e77518de30b8a4223695aba98affbc0c43473

LIB-01: If Else Instead Of Match

Type	Severity	Location
Language Specific	Informational	lib.rs L39

Description:

The code uses an if else pattern instead of match.

Recommendation:

Consider rewriting this using match.

Alleviation:

The team opted to not alleviate the issue in this iteration.

ERR-01: Inefficient Parameter

Type	Severity	Location
Logical Issue	Informational	errors.rs L38-L40

Description:

The `Schedule::err_pool_not_found` function takes an owned `name: String` parameter, which is inefficient.

Recommendation:

Consider changing the type of the `name` parameter to `&str` and removing the reference bind (`&`) on L40.

Alleviation:

The team has fixed the issue in commit f72eb69a11be11bc39c84144e0dc7a3269e9c090.

LIB-02: Unclear Implementation Naming

Type	Severity	Location
Logical Issue	Informational	lib.rs L47-L49

Description:

The `Schedule::subtotal` function has both an unclear name and implementation.

Recommendation:

Consider renaming it to `Schedule::calculate_total` and replacing the `fold` with a simplified `map` / `sum` approach.

Alleviation:

The team fixed the issue in commit up to 0a52dd41b081953702585f9e2cdc6e4033f6a15e

LIB-03: Code Structure

Type	Severity	Location
Logical Issue	Informational	lib.rs L65-L68

Description:

The code design could be implemented in a more optimal way.

Recommendation:

The `Pool::partial` function can be rewritten to be contained within the `Pool` constructor statement

Alleviation:

The team fixed the issue in commit up to 0a52dd41b081953702585f9e2cdc6e4033f6a15e

LIB-04: Code Structure

Type	Severity	Location
Logical Issue	Informational	lib.rs L69-L74

Description:

The code design could be implemented in a more optimal way.

Recommendation:

The `Pool::full` function can be rewritten to be contained within the `Pool` constructor statement

Alleviation:

The team fixed the issue in commit up to 0a52dd41b081953702585f9e2cdc6e4033f6a15e

LIB-05: Unclear Implementation Naming

Type	Severity	Location
Logical Issue	Informational	lib.rs L76-78

Description:

The `Pool::subtotal` function has both an unclear name and implementation.

Recommendation:

Consider renaming it to `Pool::calculate_total` and replacing the `fold` with a simplified `map` / `sum` approach.

Alleviation:

The team fixed the issue in commit up to 0a52dd41b081953702585f9e2cdc6e4033f6a15e

MUT-01: Inefficient Parameter

Type	Severity	Location
Logical Issue	Informational	mutate.rs L6, L12

Description:

The `Schedule::add_account` function takes a `pool_name: String` parameter on L6, which is inefficient because it is only set up this way so that `pool_name` can be passed by-value to the `Schedule::err_pool_not_found` function on L12. More often than not, this function should be expected to succeed.

Recommendation:

It would be better to pass a static string slice (`&str`) instead of an already-owned `String` that has dynamically-allocated memory. Consider changing the type of the `pool_name` parameter to `&str` on L6.

Alleviation:

The team has fixed the issue in commit e003db3be781aa2d2733eacf788518d2579128f6.

MUT-02: Unnecesary Return

Type	Severity	Location
Logical Issue	Informational	mutate.rs L28-L29

Description:

The `Pool::add_account` function returns an explicit `Ok(())` statement on L29 after making a call to the `Pool::validate` function on L28, which is unnecessary.

Recommendation:

Consider removing the semicolon from the end of L28 and removing the `Ok(())` statement on L29 altogether in order to allow the `UsuallyOk` result from `Pool::validate` to be used as the `UsuallyOk` result of the `Pool::add_account` function.

Alleviation:

The team has fixed the issue in commit b0c8dc9666aaeaaafbe661c2f91122f37ab54c50.

VAL-01: Implement Validation Trait

Type	Severity	Location
Logical Issue	Informational	validate.rs L23-L25, L34-L36

Description:

The `Schedule::validate` and `Pool::validate` functions perform loops over `Vec` elements which also implement the `Validation` trait, short-circuiting in case of failure.

Recommendation:

Consider implementing the `Validation` trait for `Vec` so that values of type `Vec<dyn Validate>` can have the `.validate()` function called on them directly.

Alleviation:

The team has fixed the issue in commit 83ee675c04f30de228be28dc20c2fd0754bda958.

VES-01: Error Handling Implementation Missing

Type	Severity	Location
Logical Issue	Medium	vesting.rs L5, L7

Description:

The `Vesting` trait requires that the generic `A` type implements both `Clone` and `PartialEq` , which is unnecessary. Implementors can restrict the generic type `A` in their own scope if necessary. Furthermore, the `Vesting::unlocked` function returns a `u128` and does not leave room for error, which leaves implementors with no choice other than to panic.

Recommendation:

Consider removing the restrictions on the generic type `A` and returning `StdResult<u128>` for the `Vesting::unlocked` function instead.

Alleviation:

The team has fixed the issue in commit 21086f55a83c6549b951bbda92e6b346d852ecad and 729e9b87b2b7db1b46c1884e8dd4341a0d2700f7.

VES-02: Implement Unlock Trait

Type	Severity	Location
Logical Issue	Informational	vesting.rs L12, L18

Description:

The `Schedule::unlocked` and `Pool::unlocked` functions perform loops over `Vec` elements which also implement the `Vesting` trait, short-circuiting in case of failure.

Recommendation:

Consider implementing the `Vesting` trait for `Vec` so that values of type `Vec` can have the `.unlocked(elapsed, address)` function called on them directly.

Alleviation:

The team fixed the issue in commit up to 0a52dd41b081953702585f9e2cdc6e4033f6a15e

VES-03: Simplification On Match Pattern

Type	Severity	Location
Logical Issue	Informational	vesting.rs L75-L84

Description:

The pattern matching of the Account::most_recent_portion function can be simplified using Option::map.

Recommendation:

Consider using Option::map.

Alleviation:

The team has fixed the issue in commit 92d02c58896b24a992d8c4c76da8a0588e0a673d.

VIE-01: Hard Coded Value

Type	Severity	Location
Language Specific	Informational	viewing_key.rs L39

Description:

The code uses a hardcoded value accomodated by a comment.

Recommendation:

Consider making it a const variable.

Alleviation:

The team has fixed the issue in commit 656d6cfb00c06d72f8cfa687b9e95ac65559ef7e.

Appendix

Finding Categories

Logical Issue

Logical Issue findings are exhibits that detail a fault in the logic of the linked code.

Language Specific

Language Specific findings are issues that would only arise within Rust.

Coding Style

Coding Style findings usually do not affect the generated byte-code and comment on how to make the codebase more legible and as a result easily maintainable.