Code Security Assessment

iSwap 7

Mar 3rd, 2022

Table of Contents

Summary

Overview

- Project Summary
- Audit Summary
- Vulnerability Summary
- Audit Scope

Findings

- BCP-01 : Centralization Related Risks
- BCP-02 : Centralization Related Risks - Token Withdrawal
- BCP-03 : Missing Input Validation
- BCP-04 : Incorrect Parameters Deserialize
- BCP-05 : Incomplete `cross_chain_fee` check
- BCP-06 : Potential Overflow
- BCP-07 : Lack of `mut` Constraint
- BCP-08 : Third Party Dependencies
- BCP-09 : Typo
- BCP-10 : Missing Emit Events

Appendix

Disclaimer

About

Summary

This report has been prepared for Iswap to discover issues and vulnerabilities in the source code of the iSwap 7 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:

- 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.

Overview

Project Summary

Project Name	iSwap 7
Platform	Solana
Language	Rust
Codebase	Private Codebase
Commit	sha256 value of zip source files: a77c0c1b22416a8ffce1af4606844162f4465f62a87ee9ea33065c30192ca2b4, 50a8d3742071ac7a00c831b6459850daab1b721703c83ad7879a5f12f569e9b8

Audit Summary

Delivery Date	Mar 03, 2022
Audit Methodology	Manual Review, Static Analysis

Vulnerability Summary

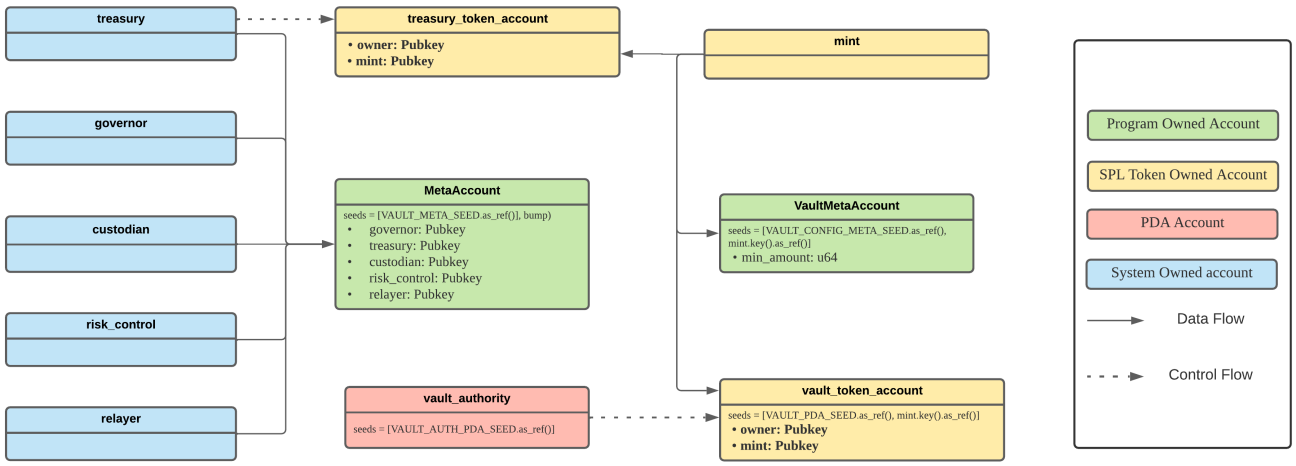
Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Mitigated	Resolved
Critical	0	0	0	0	0	0	0
Major	2	0	0	2	0	0	0
Medium	2	0	0	1	0	0	1
Minor	4	0	0	2	0	0	2
Informational	2	0	0	1	0	0	1
Discussion	0	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
BCP	src/lib.rs	a27001fa8f10c48818c40a09db3e561ba00dbaecc661769f87bebb0777ab4841
CBC	Cargo.toml	dc67c8f4bbea95447ca2a67e3406e5b949a6a4c83aa01efa4acfe78754f21486

Understandings

System overview



External Dependencies

The project mainly contains the following dependencies:

Dependency	Version
anchor-lang	0.19.0
anchor-spl	0.19.0
spl-associated-token-account	1.0.3
spl-token	3.1.1

It is important to note that the current audit scope only includes the bridge implementation on the Solana side, which implements the token transfer flow between the vault, user, and treasury accounts. The verification of the message from the other chain is processed by the **relayer** component, whose implementation is unknown. We treat the **relayer** component of the project as a black box and assume it is functionally correct.

It should also be noted here that the code dependencies are actively developed in the current auditing version. It is necessary to keep the dependencies up-to-date to avoid potential vulnerabilities.

The on-chain program can be upgradeable after the initial deployment based on Solana's features. Also, based on the unique rent mechanism in Solana, the balance in the account should be carefully set.

We assume these dependencies are valid and non-vulnerable factors and implement proper logic to collaborate with the current project.

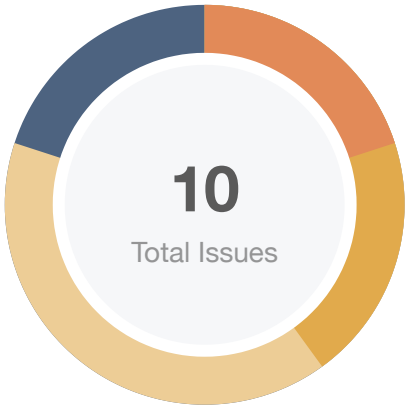
Privileged Functions

The program `i_bridge_solana_program` contains the following privileged functions/instructions that are restricted by multiple roles, and they are used to modify the contract configurations and address attributes:

- The **`governer`** can:
  - `change_governor()` will assign a new account as `governor`.
  - `change_treasury()` will assign a new account as `treasury`.
  - `change_custodian()` will assign a new account as `custodian`.
  - `change_risk_control()` will assign a new account as `risk_control`.
  - `change_relayer()` will assign a new account as `relayer`.
  - `add_support_token()` will add supported token.
  - `set_support_token_min_amount` will set min amount for one supported token.
- The **`relayer`** can withdraw an arbitrary amount of tokens from the vault token account via the function `refund_token` and `cross_chain_token_confirm`.
- The **`custodian`** can withdraw an arbitrary amount of tokens from the vault token account via function `withdrawal`.
- The **`risk_control`** can withdraw an arbitrary amount of tokens from the vault token account via function `withdrawal_punish`.
- The **`treasury`** role is the owner of the treasury token account, it can withdraw tokens from the treasury token account.

To improve the trustworthiness of the project, dynamic runtime updates in the project should be notified to the community.

Findings



Critical	**0** (0.00%)
Major	**2** (20.00%)
Medium	**2** (20.00%)
Minor	**4** (40.00%)
Informational	**2** (20.00%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
BCP-01	Centralization Related Risks	**Centralization / Privilege**	**Major**	Acknowledged
BCP-02	Centralization Related Risks - Token Withdrawal	**Centralization / Privilege**	**Major**	Acknowledged
BCP-03	Missing Input Validation	Volatile Code	Medium	Acknowledged
BCP-04	Incorrect Parameters Deserialize	Language Specific	Medium	Resolved
BCP-05	Incomplete `cross_chain_fee` check	Logical Issue	Minor	Acknowledged
BCP-06	Potential Overflow	Mathematical Operations	Minor	Resolved
BCP-07	Lack of `mut` Constraint	Language Specific	Minor	Resolved
BCP-08	Third Party Dependencies	Volatile Code	Minor	Acknowledged
BCP-09	Typo	Coding Style	Informational	Resolved
BCP-10	Missing Emit Events	Coding Style	Informational	Acknowledged

BCP-01 | Centralization Related Risks

Category	Severity	Location	Status
Centralization / Privilege	**Major**	src/lib.rs: 281, 252, 207, 163, 72, 64, 58, 52, 46, 40, 79	Acknowledged

Description

In the program `i_bridge_solana_program`, the role `governor` has authority over the following functions:

- `change_governor()` will assign a new account as `governor`.
- `change_treasury()` will assign a new account as `treasury`.
- `change_custodian()` will assign a new account as `custodian`.
- `change_risk_control()` will assign a new account as `risk_control`.
- `change_relayer()` will assign a new account as `relayer`.
- `add_support_token()` will add supported token.
- `set_support_token_min_amount` will set min amount for one supported token.

Any compromise to the `governor` account may allow the hacker to take advantage of this and result in unexpected loss.

Recommendation

These centralization-related risks described in the current project potentially need multiple iterations to improve in the security operation and level of decentralization, and in most cases can't be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's keypair 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.

Noted: Recommend considering the long-term solution or the permanent solution. The project team shall make a decision based on the current state of their project, timeline, and project resources.

Alleviation

[iSwap]: We have a strict private key protection mechanism (signature machine mechanism), which is difficult to be attacked.

1. The private key is generated, encrypted, and stored by multiple parties. It can be accessed only after each party is authorized one by one;
2. During the signing process, we need to obtain partial private keys from multiple parties by encrypted transmission, and assemble them into the private key to complete the signature;
3. Hackers need to break the firewall first, then break multiple parties. Hackers need to break several symmetric and asymmetric encryption algorithms during this process, which is extremely difficult.

We have a strong emergency response mechanism:

1. We do not issue tokens and there are no user lock-ups, so user assets will not suffer losses;
2. We have a complete monitoring mechanism. When abnormal conditions such as the abnormal decrease of assets, loss of cross-chain transactions, inconsistent request hashes, etc. occur, the problem can be found within five minutes and resolved in time.

BCP-02 | Centralization Related Risks - Token Withdrawal

Category	Severity	Location	Status
Centralization / Privilege	**Major**	src/lib.rs: 207	ⓘ Acknowledged

Description

To transfer tokens to another chain, users need to deposit tokens to the vault token account and treasury token account. However, those tokens can be withdrawn by privileged roles.

- The `relayer` can withdraw an arbitrary amount of tokens from the vault token account via the function `refund_token` and `cross_chain_token_confirm`.
- The `custodian` can withdraw an arbitrary amount of tokens from the vault token account via function `withdrawal`.
- The `risk_control` can withdraw an arbitrary amount of tokens from the vault token account via function `withdrawal_punish`.
- The `treasury` role is the owner of the treasury token account, it can withdraw tokens from the treasury token account.

The concern is, if those privileged accounts are compromised, it could lead to the asset being stolen and thus introducing centralization risk,

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 advise the client to carefully manage the privileged role's keypair 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.

Noted: Recommend considering the long-term solution or the permanent solution. The project team shall make a decision based on the current state of their project, timeline, and project resources.

Alleviation

[iSwap]: The principle of cross-chain is that the user transfers funds into the source chain contract, and the target chain transfers to the user. The liquidity of all chains is provided by the project. So the funds need to be withdrawn when the subsequent upgrade is required, that is why the withdrawal function exists. The Treasury account is the fee income account, and users' cross-chain fee will be transferred to this account, which will serve as the income source of the project.

BCP-03 | Missing Input Validation

Category	Severity	Location	Status
Volatile Code	Medium	src/lib.rs: 88, 90~92	Acknowledged

Description

To transfer tokens to another blockchain, users need to deposit a certain amount of tokens to the vault account via the function `cross_chain_token`. However, this function lacks basic verifications for users' input, thus increasing the attack vector.

```
86      pub fn cross_chain_token(
87          ctx: Context<CrossChainToken>,
88          order_id: u64,
89          amount: u64,
90          cross_chain_fee: u64,
91          gas_fee: u64,
92          rewards: u64,
93          dst_chain_id: u64,
94          channel: String,
95      ) -> ProgramResult {
```

The function `cross_chain_token` takes `order_id`, `cross_chain_fee`, `gas_fee` and `rewards` as inputs, and does not verify their validity.

- `order_id`
  This input value `order_id` would be emitted in the Event as a notification to the relayer. As there is no validation for the `order_id`, the `order_id` can be forged (e.g., input a previous `order_id` that deposits a great number of tokens). If the relayer's verification process fails, the attacker might claim others' orders.
- `cross_chain_fee`
  `cross_chain_fee` is the amount of fee that is required by the project. However, the user can input `cross_chain_fee` as 0 to bypass the charging of the fee.
- `gas_fee`
  `gas_fee` is the amount of gas fee that is required by the project. However, the user can input `gas_fee` as 0 to bypass the charging of the fee.
- `rewards`
  `rewards` will be emitted in the Event as a notification, which specified the amount of reward that can

be claimed in the target chain. The attacker can input `rewards` as a big number to get a great number of rewards in the target chain.

Recommendation

Recommend validating the aforementioned inputs injected by users. For example,

For `cross_chain_fee`, `gas_fee` and `rewards`, the inputs should be within a certain range.

For `order_id`, in the short term, the contract needs to verify if it is the order_id is valid ot not. In the long term, it is recommended to generate `order_id` within the contract instead of provided by users.

Alleviation

[iSwap]: The verification of parameters is controlled by the off-chain program, which will assign `orderId`, calculate `cross_chain_fee`, `gas_fee`, etc., and record them in the database, and compare them with the events thrown by the user after sending the transaction. If the user modifies the parameters, it will be considered malicious behavior and will be punished accordingly. Including but not limited to locked funds, frozen funds.

BCP-04 | Incorrect Parameters Deserialize

Category	Severity	Location	Status
Language Specific	Medium	src/lib.rs: 469, 505, 541	Resolved

Description

In Anchor framework, instruction's arguments can be accessed with the `#[instruction(..)]` attribute, with the same order as in the instruction, and arguments after the last needed argument can be omitted.

In instruction `cross_chain_token`, arguments are `order_id:u64`, `amount:u64`, `cross_chain_fee:u64`, `gas_fee:u64`, `rewards:u64`, `dst_chain_id:u64`, `channel:String` in order. And in `#[instruction(..)]` attribute, only the first u64 needed as `amount`, but it refers to `order_id:u64`.

In instruction `cross_chain_token_confirm`, arguments are `order_id:u64`, `amount:u64`, `rewards:u64` in order. And in `#[instruction(..)]` attribute, only the first u64 needed as `amount`, but it refers to `order_id:u64`.

In instruction `refund_token`, arguments are `order_id:u64`, `amount:u64`, `gas_fee:u64` in order. And in `#[instruction(..)]` attribute, only the first u64 needed as `amount`, but it refers to `order_id:u64`.

This finding may cause chaos during instruction execution, for example, when `order_id` is a large number and the amount for `token_valut` is small, will lead the instruction to fail.

Reference: instruction-attribute

Recommendation

Recommend reordering the arguments for aforementioned instructions to make sure the argument in `#[instruction(..)]` attribute is correct.

Alleviation

[iSwap]: The team heeded the advice and resolved this issue. The modifications are reflected in the provided (zip) source file whose sha256 value is 50a8d3742071ac7a00c831b6459850daab1b721703c83ad7879a5f12f569e9b8.

BCP-05 | Incomplete `cross_chain_fee` Check

Category	Severity	Location	Status
Logical Issue	Minor	src/lib.rs: 96	Acknowledged

Description

In function `cross_chain_token`, the `cross_chain_fee` is limited by a upper-bound `amount * DENOMINATOR / CROSS_CHAIN_FEE_RATIO`, but there is no lower-bound for `cross_chain_fee`, which means the user can set fee to 0, this may cause the fee loss.

Recommendation

Recommend adding lower-bound for `cross_chain_fee` to avoid unexpected loss.

Alleviation

[iSwap]: The contract only controls the upper limit of the fee rate. The lower limit is controlled by the off-chain program, and it may be zero according to the business demand.

BCP-06 | Potential Overflow

Category	Severity	Location	Status
Mathematical Operations	Minor	src/lib.rs: 96	Resolved

Description

The mathematic operation in L96 has the potential to lead to overflow issues during runtime as the expression uses raw arithmetic "*" and "/".

```
96          if cross_chain_fee > (amount * DENOMINATOR / CROSS_CHAIN_FEE_RATIO) {
```

Recommendation

Recommend using `checked` method from Rust, for example, `checked_mul()` and `checked_div()` to avoid the potential occurrence of integer overflow.

Alleviation

[iSwap]: The team heeded the advice and resolved this issue. The modifications are reflected in the provided (zip) source file whose sha256 value is 50a8d3742071ac7a00c831b6459850daab1b721703c83ad7879a5f12f569e9b8.

BCP-07 | Lack Of `mut` Constraint

Category	Severity	Location	Status
Language Specific	Minor	src/lib.rs: 410, 419	Resolved

Description

According to the Rust doc for `anchor-lang`, the constraint `mut` is used for checking the given account is mutable and makes anchor persist any state changes.

The following accounts' states are changing but lack the `mut` constraint:

- struct `Initialize`
  - `user` will be the payer for `meta_account`.

Reference:

- https://docs.rs/anchor-lang/latest/anchor_lang/derive.Accounts.html

Recommendation

Recommend adding the constraint `mut` to the aforementioned account for improving the account constraint readability and avoid unexpected errors.

Alleviation

[iSwap]: The team heeded the advice and resolved this issue. The modifications are reflected in the provided (zip) source file whose sha256 value is 50a8d3742071ac7a00c831b6459850daab1b721703c83ad7879a5f12f569e9b8.

BCP-08 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	Minor	src/lib.rs: 163, 207	Acknowledged

Description

The contract is serving as the underlying entity to interact with the **relayer** component. The scope of the audit treats the **relayer** as black boxes and assumes their functional correctness. Additionally, the core verification process that validates the message from the other chain is implemented in the **relayer**. The **relayer** is responsible for the verification and proper invocation of functions `cross_chain_token_confirm` and `refund_token`.

Recommendation

The team should ensure the relayer component work properly as expected.

Alleviation

[iSwap]: We will pay attention when we add 3rd-party DEX and tool, their security will be checked.

BCP-09 | Typo

Category	Severity	Location	Status
Coding Style	Informational	src/lib.rs: 96, 18~19	Resolved

Description

```
96          if cross_chain_fee > (amount * DENOMINATOR / CROSS_CHAIN_FEE_RATIO) {
97              return Err(ErrorCode::ExceedTheCrossChainFeeLimit.into());
98          }
```

According to the above calculation in the `if` branch, there are two typos in the const `DENOMINATOR` and `CROSS_CHAIN_FEE_RATIO`, the const `DENOMINATOR` and `CROSS_CHAIN_FEE_RATIO` should be renamed as `CROSS_CHAIN_FEE_RATIO` and `DENOMINATOR`.

Recommendation

Consider refactoring the const names.

Alleviation

[iSwap]: The team heeded the advice and resolved this issue. The modifications are reflected in the provided (zip) source file whose sha256 value is 50a8d3742071ac7a00c831b6459850daab1b721703c83ad7879a5f12f569e9b8.

BCP-10 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	src/lib.rs: 40, 46, 52, 58, 64, 72, 22, 79	Acknowledged

Description

The functions that affect the status of sensitive variables should be able to emit events.

- `initialize`
- `change_governor`
- `change_treasury`
- `change_custodian`
- `change_risk_control`
- `change_relayer`
- `add_support_token`
- `set_support_token_min_amount`

Recommendation

Consider adding events for sensitive actions, and emit them in the functions.

Alleviation

[iSwap]: These functions don't need event listening.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

Coding Style

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

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

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.

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.