Security Assessment

Arbix Finance

Nov 19th, 2021

Table of Contents

Summary

This report has been prepared for Arbix Finance to discover issues and vulnerabilities in the source code of the Arbix Finance project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review 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	Arbix Finance
Description	ERC20 and Strategy
Platform	bsc
Language	Solidity
Codebase	5560a6af685a99b3710d228a769ac94adbe58389
Commit	5560a6af685a99b3710d228a769ac94adbe58389

Audit Summary

Delivery Date	Nov 19, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

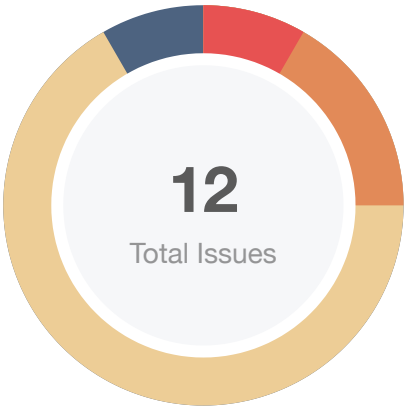
Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	1	0	0	0	0	1
Major	2	0	0	0	0	2
Medium	0	0	0	0	0	0
Minor	8	0	0	3	0	5
Informational	1	0	0	0	0	1
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
AYS	contracts/ada/AdaYieldStrategy.sol	beef9963496293221531 66ccfd5cb50f25e9b5ebaf6265ab9cf9c619266c259f
AYT	contracts/ada/AdaYieldStrategyToken.sol	f50cfbacdbc098bb4a9367ee5dc946ecc5994e02bcb8d30fdb59f4ec2baddb79
BYS	contracts/btc/BtcYieldStrategy.sol	13d11aa93ef4442d483e9e3b8a1b12c4f84a438736c14dbd7e950890537696c6
BYT	contracts/btc/BtcYieldStrategyToken.sol	a8799d9d215b6c0df8f66d66f18633078c8c171c841c6f8f8a23f6a6bdc78190
BUD	contracts/busd/BusdYieldStrategy.sol	f843e86748f9f12453f5244aadc5e43343e62dca15888fa1060ac9609705906f
BST	contracts/busd/BusdYieldStrategyToken.sol	96e388a31ee4cb8780ee85aec66418a3406d610df6e3b6a20b79c2a660704368
DYS	contracts/doge/DogeYieldStrategy.sol	33ad2ef41e6c78ac73e5768855122d10a8e7455db471dfe3fb866ac7c3018508
DYT	contracts/doge/DogeYieldStrategyToken.sol	fe68b609fa4424b666f536fb8b50ba45f28971933effdf05e639383fa5ad4fa7
DOY	contracts/dot/DotYieldStrategy.sol	d4929ce71e834985df9beefe453fc093da7a8561d573cf4dbfff3ba7fbf9068a
DST	contracts/dot/DotYieldStrategyToken.sol	64b0ea2f8c8df95e14a05419ee561836200f318f0dab8a69b6a1180a4e37d6a8
EYS	contracts/eth/EthYieldStrategy.sol	de50f3ebab6d52fa5d4a7399613f2d5fb5fc1dc9b3dd352521895aa2e561ad1e
EYT	contracts/eth/EthYieldStrategyToken.sol	2399b961127d2517aaf80e66037a3ee9cdb52315ce94a82d3674f2f56dff9c1e
LYS	contracts/link/LinkYieldStrategy.sol	07c3838eba875ac6d37bf59ce99d6af82c16a05e15f57a9f57eb893732ae99e6
LYT	contracts/link/LinkYieldStrategyToken.sol	080c934da89db5f716f1b998b3e7cae6b4c81fb32f57dfb12ceff3b25b7373b2
UYS	contracts/usdc/UsdcYieldStrategy.sol	202db426c58c0ab435498632aa9c41236d48b3d23a0e649db97c334df5f09a84
UYT	contracts/usdc/UsdcYieldStrategyToken.sol	4d9656105ddc29a315ddbe183dd7e1208fede714c19e8195070a8233e88a0896

ID	File	SHA256 Checksum
USY	contracts/usdt/UsdtYieldStrategy.sol	de4f7ba290b9d7f20b1ea265993791daf4fca987c72cc5366bb6cc69f6032f33
YST	contracts/usdt/UsdtYieldStrategyToken.sol	1f10ac2f9c0d242c3dd33d5e642b742722267bec9467ad9c667e95fd6f91c5f0
XYS	contracts/xrp/XrpYieldStrategy.sol	d5151684b3ccffc69c5b57a44b0ed44754ad6a291792522ba95285c595226962
XYT	contracts/xrp/XrpYieldStrategyToken.sol	7427ca7041fb9533f54d38c3c14118f1fe33a5e246394c12ad3c7c003c0a51a1

Findings



Critical	1 (8.33%)
Major	2 (16.67%)
Medium	0 (0.00%)
Minor	8 (66.67%)
Informational	1 (8.33%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	`govAddress` Can Withdraw `vTokenAddress` Tokens From Contract	Centralization / Privilege	Critical	Resolved
GLOBAL-02	`strategy` Requires Manual Setting And Can Be Altered By Owner	Centralization / Privilege, Volatile Code	Major	Resolved
GLOBAL-03	Centralization Risk	Centralization / Privilege	Major	Resolved
GLOBAL-04	Potential Sandwich Attacks	Logical Issue	Minor	Acknowledged
GLOBAL-05	Unknown implementations	Volatile Code	Minor	Acknowledged
GLOBAL-06	Third Party Dependencies	Volatile Code	Minor	Acknowledged
GLOBAL-07	Declare Variable as Immutable	Gas Optimization	Minor	Resolved
GLOBAL-08	Return Value Not Handled	Volatile Code	Minor	Resolved

ID	Title	Category	Severity	Status
GLOBAL-09	Scope of Code	Gas Optimization, Volatile Code	Minor	Resolved
GLOBAL-10	Missing Zero Address Validation	Volatile Code	Minor	Resolved
GLOBAL-11	BNB Locked In Contract	Logical Issue, Control Flow	Minor	Resolved
GLOBAL-12	Missing Emit Events	Coding Style	Informational	Resolved

GLOBAL-01 | `govAddress` Can Withdraw `vTokenAddress` Tokens From Contract

Category	Severity	Location	Status
Centralization / Privilege	Critical	Global	Resolved

Description

In the `StrategyVenus` contracts, the `govAddress` can withdraw `vTokenAddress` tokens from the contract via the `inCaseTokensGetStuck()` function, as the `require` statements only check that `_token` does not equal to `xvsAddress` and `underlyingAddress`, but not `vTokenAddress`. Anyone who compromises the `govAddress` account can drain all `vTokenAddreess` tokens from the contract.

```
    function inCaseTokensGetStuck(
        address _token,
        uint256 _amount,
        address _to
    ) public {
        require(msg.sender == govAddress, "!gov");
        require(_token != xvsAddress, "!safe");
        require(_token != underlyingAddress, "!safe");

        IERC20(_token).safeTransfer(_to, _amount);
    }
```

Recommendation

We recommend adding the following `require` statement to this function:

```
require(_token != vTokenAddress, "!safe");
```

Alleviation

[Arbix Team]: Resolved as suggested in the audit at the code line 1362

GLOBAL-02 | `strategy` Requires Manual Setting And Can Be Altered By Owner

Category	Severity	Location	Status
Centralization / Privilege, Volatile Code	Major	Global	Resolved

Description

In the `StrategyVenusToken` contracts, the variable `strategy` is not instantiated in the constructer requires manual setting. The implementation of the `strategy` contract could be unknown changed over time and affect critical functions such as `deposit()` and `withdraw()` which handle user transactions.

Recommendation

We recommend instantiating the `strategy` variable in the constructor and removing it's mutability.

```
constructor (
    string memory name_,
    string memory symbol_,
    address _token,
    address _underlyingAddress, //e.g. BTCB
    address _uniRouterAddress,
    address _vTokenAddress
) public ERC20(name_, symbol_) {
    token = _token; //underlyingToken
    govAddress = msg.sender;

    strategy = new StrategyVenus(_underlyingAddress, _uniRouterAddress, _vTokenAddress);

    ERC20._setupDecimals(
        ERC20(_token).decimals()
    );
}
```

Alleviation

[Arbix Team]: Resolved as suggested in the audit at the code line 1433 plus some corresponding changes in constructor arguments and the removal of the setNewStrategy function.

GLOBAL-03 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	Global	Resolved

Description

In the `StrategyVenus` contracts, the role `owner()` has the authority over the following functions:

- `renounceOwnership()`
- `transferOwnership()`
- `deposit()`
- `withdraw()`

and is assigned to `msg.sender` during construction

In the `StrategyVenus` contracts, the role `govAddress` has the authority over the following functions:

- `pause()`
- `unpause()`
- `setGov()`

and is assigned to `msg.sender` during construction

In the `StrategyVenusToken` contracts, the role `owner()` has the authority over the following functions:

- `renounceOwnership()`
- `transferOwnership()`
- `pauseDeposit()`
- `unpauseDeposit()`
- `pauseWithdraw()`
- `unpauseWithdraw()`
- `inCaseTokensGetStuck()`
- `setNewStrategy()`

and is assigned to `msg.sender` during construction

In the `StrategyVenusToken` contracts, the role `govAddress` has the authority over the following functions:

- `pauseDeposit()`

- `unpauseDeposit()`
- `pauseWithdraw()`
- `unpauseWithdraw()`
- `inCaseTokensGetStuck()`
- `setNewStrategy()`

and is assigned to `msg.sender` during construction

Any compromise to the `owner()` or `govAddress` accounts may allow the hacker to take advantage of these functions.

Recommendation

We advise the client to carefully manage the `owner()` and `govAddress` accounts's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

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

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Alleviation

[Arbix Team]: assigning privileged roles to a smart contract as multi-signature wallets powered by Gnosis-Safe. The authority of the sensitive functions is at the expense of 2-out-of-3 privileged wallet proposal votes (by Arbix's personnels) - effectively resolving single-point-of-failure issue. Resolved as suggested in the audit at the code line 1427 plus some corresponding changes in constructor arguments

Deployed Contracts Bsc Mainnet (Chain Id: 56)

- AdaYieldStrategyToken: 0xfFEC7af8E39cdA29395B55e658fB7193797ccD60
- BusdYieldStrategyToken: 0xDCDE38d6072144813e39a2850e79f784E6B798F7
- DogeYieldStrategyToken: 0xFA0458E298b2d398d214D1aE03fEB5A57b39b559
- DotYieldStrategyToken: 0xc11776D07131ceFaE61b8b23e16317c2bcd18093
- EthYieldStrategyToken: 0x6348fe4A573eaBDd3F30fDcb1b80B05aC6871574
- LinkYieldStrategyToken: 0x3D57AF470fC9BE449e12f676B8230ed790A844bF

- UsdcYieldStrategyToken: 0x611b8849655955be0D7035e590B61eD3FC2eE2f8
- UsdtYieldStrategyToken: 0x43ae3ED3317EFcef7faFae3068F0Bd8966577810
- VbtcYieldStrategyToken: 0x6863a5De6554222313bEbFfE69d2F5543D2450dB
- XrpYieldStrategyToken: 0xDeE259536b59523077cA2c0120bfF43CC33AA24B

GLOBAL-04 | Potential Sandwich Attacks

Category	Severity	Location	Status
Logical Issue	Minor	Global	Acknowledged

Description

A sandwich attack might happen when an attacker observes a transaction swapping tokens or adding liquidity without setting restrictions on slippage or minimum output amount. The attacker can manipulate the exchange rate by frontrunning (before the transaction being attacked) a transaction to purchase one of the assets and make profits by backrunning (after the transaction being attacked) a transaction to sell the asset.

In the `StrategyVenus` contracts, the following function is called without setting restrictions on slippage or minimum output amount, so transactions triggering these functions are vulnerable to sandwich attacks, especially when the input amount is large:

- `swapExactTokensForTokens()`

Recommendation

We recommend setting reasonable minimum output amounts, instead of 0, based on token prices when calling the aforementioned functions.

Alleviation

[Arbix Team]: This function is used only for calling earn(). The earn() function will be calling so frequently the swap amount each time would be so small it wouldn't be worth the attacker to perform the attack.

GLOBAL-05 | Unknown implementations

Category	Severity	Location	Status
Volatile Code	Minor	Global	Acknowledged

Description

In the `StrategyVenus` contracts, the interface `IVenusToken` is an unknown implementation. The scope of the audit treats 3rd party entities as black boxes and assume their functional correctness. However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of 3rd parties can possibly create severe impacts, such as increasing fees of 3rd parties, migrating to new LP pools, etc.

Alleviation

[Arbix Team]: IVenusToken is an interface to Venus Protocol's VBep20. Its functionalities are to access Venus's liquidity pool by converting the underlying token into a corresponding Venus token.

GLOBAL-06 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	Minor	Global	Acknowledged

Description

The contract is serving as the underlying entity to interact with third party PancakeSwap and Venus protocols. The scope of the audit treats 3rd party entities as black boxes and assume their functional correctness. However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of 3rd parties can possibly create severe impacts, such as increasing fees of 3rd parties, migrating to new LP pools, etc.

Recommendation

We understand that the business logic requires interaction with PancakeSwap and Venus. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

[Arbix Team]: our team will be responding to changes and perform migrations if the changes from the third party took effect.

GLOBAL-07 | Declare Variable as Immutable

Category	Severity	Location	Status
Gas Optimization	Minor	Global	Resolved

Description

In the `StrategyVenus` contracts, the variable `underlyingAddress`, `vTokenAddress` assigned in the constructor can declare with `Immutable`. In the `StrategyVenusToken` contracts, the variable `token` assigned in the constructor can declare with `Immutable`. Immutable state variables can be assigned during contract creation but will remain constant throughout the lifetime of a deployed contract. A big advantage of immutable variables is that reading them is significantly cheaper than reading from regular state variables since will not be stored in storage. Still, values will directly insert the values into the runtime code.

Recommendation

We recommend using an immutable state variable for `underlyingAddress`, `vTokenAddress`.

Alleviation

[Arbix Team]: Resolved as suggested in the audit at code lines 1214 and 1227

GLOBAL-08 | Return Value Not Handled

Category	Severity	Location	Status
Volatile Code	Minor	Global	Resolved

Description

In the `StrategyVenus` contracts, the function calls to `mint()`, `swapExactTokensForTokens()` and `redeem()` return values that should be checked.

Recommendation

We recommend using variables to receive the return value of the functions mentioned above and handle both success and failure cases if needed by the business logic.

Alleviation

[Arbix Team]: Resolved as suggested in the audit at code lines 1268, 1290, 1292, and 1308.

GLOBAL-09 | Scope of Code

Category	Severity	Location	Status
Gas Optimization, Volatile Code	Minor	Global	Resolved

Description

In the `StrategyVenus` contracts, should the `if` conditional `(xvsAddress != underlyingAddress)` not execute, the `swappedEarnedAmount` will be 0 and attempt to mint zero tokens.

Recommendation

We recommend restructuring the related code in the scope of the `if` conditional as follows:

```
  960  if (xvsAddress != underlyingAddress) {
  961    uint256 swappedEarnedAmount =
IERC20(underlyingAddress).balanceOf(address(this));
  962    IPancakeRouter02(uniRouterAddress).swapExactTokensForTokens(
  963        earnedAmt,
  964        0,
  965        xvsToUnderlyingPath,
  966        address(this),
  967        block.timestamp.add(600)
  968    );
  969    swappedEarnedAmount =
IERC20(underlyingAddress).balanceOf(address(this)).sub(swappedEarnedAmount);
  970    IVenusToken(vTokenAddress).mint(swappedEarnedAmount);
  971  }
  972
```

Alleviation

[Arbix Team]: Resolved as suggested in the audit at code lines 1282 - 1293

GLOBAL-10 | Missing Zero Address Validation

Category	Severity	Location	Status
Volatile Code	Minor	Global	Resolved

Description

In the `StrategyVenus` contracts, the function `setGov()` lacks a check on `address(0)` potentially locking out access to the associated privileged functions, including `setGov()`. This would drastically affect the operability of the contract.

Recommendation

Check that the address is not zero by adding following checks.

```
function setGov(address _govAddress) public {
    require(msg.sender == govAddress, "Not authorised");
    require(_govAddress != address(0));
    govAddress = _govAddress;
}
```

Alleviation

[Arbix Team]: Resolved as suggested in the audit at the code line 1343

GLOBAL-11 | BNB Locked In Contract

Category	Severity	Location	Status
Logical Issue, Control Flow	Minor	Global	Resolved

Description

The `StrategyVenus` and `StrategyVenusToken` contracts include `recieve()` and `fallback()` functions that capture BNB, however there is no way to withdraw BNB sent to the contract

Alleviation

[Arbix Team]: Resolved as suggested in the audit at code lines 1364 - 1369 and 1569 - 1574

GLOBAL-12 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	Global	Resolved

Description

In the `StrategyVenus` contracts, the function that affects the status of sensitive variables should be able to emit events as notifications.

- `earn()` updates `lastEarnBlock`
- `setGov()` updates `govAddress`
- `setPancakeRouterV2()` updates `uniRouterAddress`

In the `StrategyVenusToken` contracts, the function that affects the status of sensitive variables should be able to emit events as notifications.

- `setGov()` updates `govAddress`
- `_setNewStrategy()` updates `strategy`

Recommendation

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

Alleviation

[Arbix Team]:Resolved as suggested in the audit at code lines 1297, 1344, 1377, and 1440

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.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

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.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Volatile Code

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

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.