CERTIK

# Code Security Assessment

# Lobby

Jan 14th, 2022

# Table of Contents


## Summary

## Overview
- Project Summary
- Audit Summary
- Vulnerability Summary
- Audit Scope

## Findings
- LLC-01 : Centralization Related Risks
- LLC-02 : Incorrect naming convention utilization
- LLC-03 : Inaccurate Error Message
- LLC-04 : Variable Declare As Constant
- LLC-05 : Initial token distribution
- LLC-06 : The purpose of function `deliver`
- LLC-07 : Incorrect error message
- LLC-08 : Potential Sandwich Attacks
- LLC-09 : Centralized risk in `addLiquidity`
- LLC-10 : Return value not handled
- LLC-11 : Redundant code
- LLC-12 : Lack of Specified Rate Range Restriction
- LLC-13 : Missing Input Validation
- LLC-14 : Missing `require` Statements

## Appendix

## Disclaimer

## About

# Summary

This report has been prepared for Lobby to discover issues and vulnerabilities in the source code of the Lobby 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 | Lobby |
| Platform | ethereum |
| Language | Solidity |
| Codebase | https://etherscan.io/address/0xac042d9284df95cc6bd35982f6a61e3e7a6f875b |
| Commit | |

## Audit Summary

| | |
|---|---|
| Delivery Date | Jan 14, 2022 |
| Audit Methodology | Static Analysis, Manual Review |

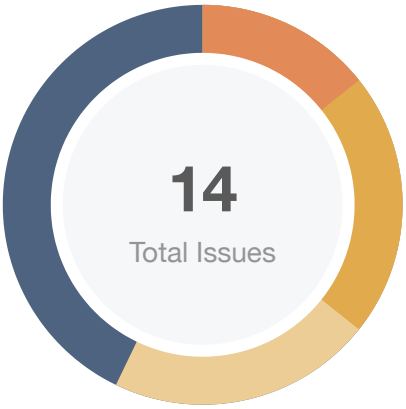
## Vulnerability Summary

| Vulnerability Level | Total | Pending | Declined | Acknowledged | Partially Resolved | Resolved |
|---|---|---|---|---|---|---|
| Critical | 0 | 0 | 0 | 0 | 0 | 0 |
| Major | 2 | 0 | 0 | 0 | 2 | 0 |
| Medium | 3 | 0 | 0 | 2 | 1 | 0 |
| Minor | 3 | 0 | 0 | 3 | 0 | 0 |
| Informational | 6 | 0 | 1 | 5 | 0 | 0 |
| Discussion | 0 | 0 | 0 | 0 | 0 | 0 |

# Audit Scope

| ID | File | SHA256 Checksum |
| --- | --- | --- |
| LLC | Lobby/project/contracts/Lobby.sol | 53416f77884ea707deeb75a1571103233ba53d872ef2a395bc12f903b3b10200 |

# Findings




| | |
|---|---|
| **Critical** | **0** (0.00%) |
| **Major** | **2** (14.29%) |
| **Medium** | **3** (21.43%) |
| **Minor** | **3** (21.43%) |
| **Informational** | **6** (42.86%) |
| **Discussion** | **0** (0.00%) |

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| **LLC-01** | Centralization Related Risks | **Centralization / Privilege** | **Major** | Partially Resolved |
| LLC-02 | Incorrect naming convention utilization | Volatile Code | Informational | Acknowledged |
| LLC-03 | Inaccurate Error Message | Coding Style | Informational | Acknowledged |
| LLC-04 | Variable Declare As Constant | Gas Optimization | Informational | Acknowledged |
| **LLC-05** | Initial token distribution | **Centralization / Privilege** | **Medium** | Partially Resolved |
| LLC-06 | The purpose of function `deliver` | Control Flow | Informational | Acknowledged |
| LLC-07 | Incorrect error message | Logical Issue | Medium | Acknowledged |
| LLC-08 | Potential Sandwich Attacks | Logical Issue | Minor | Acknowledged |
| **LLC-09** | Centralized risk in `addLiquidity` | **Centralization / Privilege** | **Major** | Partially Resolved |
| LLC-10 | Return value not handled | Volatile Code | Informational | Declined |
| LLC-11 | Redundant code | Logical Issue | Informational | Acknowledged |
| LLC-12 | Lack of Specified Rate Range Restriction | Logical Issue | Medium | Acknowledged |
| LLC-13 | Missing Input Validation | Volatile Code | Minor | Acknowledged |

| ID | Title | Category | Severity | Status |
|---|---|---|---|---|
| LLC-14 | Missing `require` Statements | Logical Issue | Minor | Acknowledged |

## LLC-01 | Centralization Related Risks

| Category | Severity | Location | Status |
|---|---|---|---|
| Centralization / Privilege | Major | Lobby/project/contracts/Lobby.sol: 460, 469, 480, 848, 860, 895, 905, 928, 932, 935, 940, 944, 949, 953, 958, 963, 968, 972, 976, 980, 993, 988 | Partially Resolved |

### Description

In the contract, the role `owner` has the authority over the following function:

- `renounceOwnership()`
- `transferOwnership()`
- `transferOwnership()`
- `airdrop()`
- `airdropArray()`
- `excludeFromReward()`
- `includeInReward()`
- `excludeFromFee()`
- `includeInFee()`
- `setCharityFeePercent()`
- `setCharityWallet()`
- `setTaxFeePercent()`
- `setLiquidityFeePercent()`
- `setMaxTxAmount()`
- `setSwapThresholdAmount()`
- `claimTokens()`
- `claimOtherTokens()`
- `clearStuckBalance()`
- `addBotWallet()`
- `removeBotWallet()`
- `setSwapAndLiquifyEnabled()`
- `allowtrading()`

Any compromise to the `owner` account may allow the hacker to take advantage of this.

### 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 account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

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

## Short Term:

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

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

## Long Term:

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

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

## Permanent:

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

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

## Alleviation

The team acknowledged the issue and adopted the multisign solution to ensure the private key management process at the current stage. The `Lobby` contract has transferred the ownership to a Gnosis Safe contract with 2/3 signers in the sensitive function signing process.

- Grant Role transaction hash for Gnosis Safe: 0xBb38A5e595990ED3Fe78e827bf77763d4DC0Acf4
- The 3 signers' addresses:
    1. EOA:0x8C49a7C4b48cCCbd998eE415E525b9E2287f23f2
    2. EOA:0x65ce4cDDF52A86B037C182FD949C1A97a30d16df
    3. EOA:0xC41de302d83Af5d76F20FCcfD5572Cde02782666

In addition, in regards to owner control, the team stated they will release a whitepaper describing each team member, their background, and involvement with the Lobby project.

The team implemented a voting module at https://vote.lobbytoken.io to increase transparency and community involvement.

## LLC-02 | Incorrect naming convention utilization

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | Informational | Lobby/project/contracts/Lobby.sol: 475, 760 | Acknowledged |

### Description

Solidity defines a naming convention that should be followed. In general, the following naming conventions should be utilized in a Solidity file.

Refer to https://solidity.readthedocs.io/en/v0.5.17/style-guide.html#naming-conventions

1. Function name is mistakenly set as `geUnlockTime()`, should be 'getUnlockTime()'.
2. In the following code snippet, `tokensIntoLiqudity` should be `tokensIntoLiquidity`.

```
757     event SwapAndLiquify(
758         uint256 tokensSwapped,
759         uint256 ethReceived,
760         uint256 tokensIntoLiqudity
761     );
```

### Recommendation

We recommend correcting all typos in the contract.

### Alleviation

The team acknowledged this issue and they will leave it as it is for now since the contract was deployed.

## LLC-03 | Inaccurate Error Message

| Category | Severity | Location | Status |
|---|---|---|---|
| Coding Style | Informational | Lobby/project/contracts/Lobby.sol: 490 | Acknowledged |

### Description

This error message does not give accurate feedback when exceptions occur.

```
490     require(now > _lockTime , "Contract is locked until 7 days");
```

### Recommendation

we recommend doing the changes as bellowed,

```
490     require(now > _lockTime , "Contract is locked until lock days");
```

### Alleviation

The team acknowledges the finding, and given the deployed contract cannot be updated, decided to retain the code base unchanged

## LLC-04 | Variable Declare As Constant

| Category | Severity | Location | Status |
|---|---|---|---|
| Gas Optimization | Informational | Lobby/project/contracts/Lobby.sol: 734~736, 723 | Acknowledged |

### Description

Variables `_name`, `_symbol`, `_decimals` and `botscantrade` could be declared as `constant` since these state variables are never to be changed.

### Recommendation

We recommend that those state variables should be declared `constant` to save gas.

### Alleviation

The team acknowledged this issue and they will leave it as it is for now since the contract was deployed.

## LLC-05 | Initial token distribution

| Category | Severity | Location | Status |
|---|---|---|---|
| **Centralization / Privilege** | **Medium** | Lobby/project/contracts/Lobby.sol: 770 | Partially Resolved |

### Description

All amounts of the tokens are sent to the contract deployer when deploying the contract. This could be a centralization risk as the deployer can distribute tokens without obtaining the consensus of the community. Since the privilege of the deployer, it is possible of being maliciously manipulated by hackers if the account of the deployer was compromised.

### Recommendation

We recommend the team be transparent regarding the initial token distribution process. Besides, we advise the client to carefully manage the project's private key to avoid any potential risks of being hacked.

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 account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

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

### Short Term:

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

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

## Long Term:

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

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

## Permanent:

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

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

## Alleviation

The team partially resolved this issue and they stated the following:

"Total circulating supply is 10 billion LBY, they set aside 1 billion LBY tokens to make up the treasury, the remaining 9 billion of circulating supply was locked in a liquidity pool for 6 months with the option to extend time-lock. And, a voting module has been implemented to increase transparency and community involvement."

## LLC-06 | The purpose of function `deliver`

| Category | Severity | Location | Status |
|---|---|---|---|
| Control Flow | Informational | Lobby/project/contracts/Lobby.sol: 869~876 | Acknowledged |

### Description

The function `deliver` can be called by anyone. It accepts an uint256 number parameter `tAmount`. The function reduces the `KAINET` token balance of the caller by `rAmount`, which is `tAmount` reduces the transaction fee. Then, the function adds `tAmount` to variable `_tFeeTotal`, which represents the contract's total transaction fee. We wish the team could explain more on the purpose of having such functionality.

### Alleviation

The team acknowledged this issue and they stated the deliver function can donate directly to the funds distributed as rewards.

## LLC-07 | Incorrect error message

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | Medium | Lobby/project/contracts/Lobby.sol: 906 | Acknowledged |

### Description

The error message in `require(_isExcluded[account], "Account is already excluded")` does not describe the error correctly.

### Alleviation

The team acknowledged this issue and they will leave it as it is for now since the contract was deployed.

## LLC-08 | Potential Sandwich Attacks

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | Minor | Lobby/project/contracts/Lobby.sol: 1182~1188, 1196~1203 | 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 back running (after the transaction being attacked) a transaction to sell the asset.

The following functions are 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:

- `uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens()`
- `uniswapV2Router.addLiquidityETH()`

### Recommendation

We advise the client to use Oracle to get an estimation of prices and setting minimum amounts based on the prices when calling the aforementioned functions.

### Alleviation

The team acknowledged this issue and they will leave it as it is for now since the contract was deployed.

## LLC-09 | Centralized risk in `addLiquidity`

| Category | Severity | Location | Status |
|---|---|---|---|
| Centralization / Privilege | Major | Lobby/project/contracts/Lobby.sol: 1201 | Partially Resolved |

### Description

```
1196 // add the liquidity
1197 uniswapV2Router.addLiquidityETH{value: ethAmount}(
1198     address(this),
1199     tokenAmount,
1200     0, // slippage is unavoidable
1201     0, // slippage is unavoidable
1202     owner(),
1203     block.timestamp
1204 );
```

The `addLiquidity` function calls the `uniswapV2Router.addLiquidityETH` function with the `to` address specified as `owner()` for acquiring the generated LP tokens from the `LBY-ETH` pool. As a result, over time the `_owner` address will accumulate a significant portion of LP tokens.If the `_owner` is an EOA (Externally Owned Account), mishandling of its private key can have devastating consequences to the project as a whole.

### Recommendation

We advise the `to` address of the `uniswapV2Router.addLiquidityETH` function call to be replaced by the contract itself, i.e. `address(this)`, and to restrict the management of the LP tokens within the scope of the contract's business logic. This will also protect the LP tokens from being stolen if the `_owner` account is compromised.

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 account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

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

## Short Term:

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

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

## Long Term:

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

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

## Permanent:

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

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

## Alleviation

The team acknowledged the issue and adopted the multisign solution to ensure the private key management process at the current stage. The `Lobby` contract has transferred the ownership to a Gnosis Safe contract with 2/3 signers in the sensitive function signing process.

- Grant Role transaction hash for Gnosis Safe: 0xBb38A5e595990ED3Fe78e827bf77763d4DC0Acf4

- The 3 signers' addresses:

    1. EOA:0x8C49a7C4b48cCCbd998eE415E525b9E2287f23f2

2. EOA:0x65ce4cDDF52A86B037C182FD949C1A97a30d16df
3. EOA:0xC41de302d83Af5d76F20FCcfD5572Cde02782666

In addition, in regards to owner control, the team stated they will release a whitepaper describing each team member, their background, and involvement with the Lobby project.

# LLC-10 | Return value not handled

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | Informational | Lobby/project/contracts/Lobby.sol: 1196~1203 | Declined |

## Description

The return values of function `addLiquidityETH` are not properly handled.

```
        uniswapV2Router.addLiquidityETH{value: ethAmount}(
            address(this),
            tokenAmount,
            0, // slippage is unavoidable
            0, // slippage is unavoidable
            owner(),
            block.timestamp
        );
```

## 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

The team declined this issue and they stated the following:

"The return values of the methods need to be analyzed in specific business scenarios, not all checks are required. Some return values are the result, which means those are just to elaborate that the functions are called, so the checks do not need to be imposed on the results. There are a series of checks already done in the process."

# LLC-11 | Redundant code

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | Informational | Lobby/project/contracts/Lobby.sol: 1224 | Acknowledged |

## Description

The condition `!_isExcluded[sender] && !_isExcluded[recipient]` can be included in `else` .

## Recommendation

The following code can be removed:

```
1223 ... else if (!_isExcluded[sender] && !_isExcluded[recipient]) {
1224     _transferStandard(sender, recipient, amount);
1225 } ...
```

## Alleviation

The team acknowledged this issue and they will leave it as it is for now since the contract was deployed.

## LLC-12 | Lack of Specified Rate Range Restriction

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | Medium | Lobby/project/contracts/Lobby.sol: 950 | Acknowledged |

### Description

The `owner` of the contract has permission to modify the fees without limitation.
Therefore, in the extreme case, that fee could be a very large amount of value, which might cause unexpected loss to the project and users.

### Recommendation

We advise the client to set a reasonable range restriction for the aforementioned states to ensure the fair distribution of the fees/tokens.

### Alleviation

The team acknowledged this issue and they will leave it as it is for now since the contract was deployed.

## LLC-13 | Missing Input Validation

| Category | Severity | Location | Status |
|---|---|---|---|
| Volatile Code | Minor | Lobby/project/contracts/Lobby.sol: 940, 968, 972 | Acknowledged |

### Description

The input value to `walletAddress` should be verified as a non-zero value to prevent being mistakenly assigned as `address(0)` in the linked functions.

### Recommendation

We advise the client to add the following check to all functions mentioned above.

```
require(walletAddress != address(0), "wallet address is zero");
```

### Alleviation

The team acknowledged this issue and they will leave it as it is for now since the contract was deployed.

## LLC-14 | Missing `require` Statements

| Category | Severity | Location | Status |
|---|---|---|---|
| Logical Issue | Minor | Lobby/project/contracts/Lobby.sol: 963 | Acknowledged |

### Description

In function `claimTokens()`, there should be an additional check to guarantee that `walletaddress` address is not zero.

### Recommendation

We recommend adding the `require` statement as below,

```
964        require(charityWallet != address(0), "charity wallet address is zero");
```

### Alleviation

The team acknowledged this issue and they will leave it as it is for now since the contract was deployed.

# 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.

CERTIK