Code Security Assessment

Tipsy Coin

Feb 23rd, 2022

Table of Contents

Summary

This report has been prepared for Tipsy Coin to discover issues and vulnerabilities in the source code of the Tipsy Coin 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	Tipsy Coin
Platform	Other
Language	Solidity
Codebase	https://github.com/TipsyDev-Mittens/TipsyCoin/commit/a0804fe44899fec533f36ccd1c69ceb2f1fa4947
Commit	a0804fe44899fec533f36ccd1c69ceb2f1fa4947

Audit Summary

Delivery Date	Feb 23, 2022
Audit Methodology	Static Analysis, Manual Review

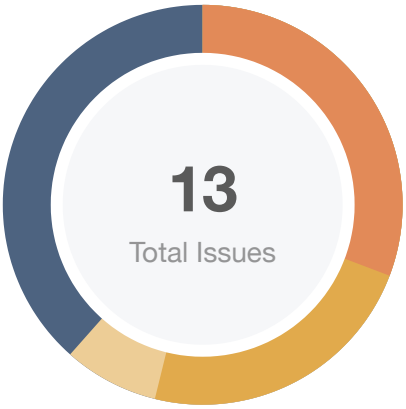
Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Mitigated	Resolved
Critical	0	0	0	0	0	0	0
Major	4	0	0	0	0	2	2
Medium	3	0	0	0	0	0	3
Minor	1	0	0	0	0	0	1
Informational	5	0	0	1	0	0	4
Discussion	0	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
BBT	TipsyCoin/contracts/BuyBack.sol	e6988470b8c359f8dcf503cc13c1f8c7a4493e8fb8f7db5c73578ca72bbe3e23
DTC	TipsyCoin/contracts/Distribution.sol	071c1d597f3f917ed437ad9072d742b99453c89d60484655917f521e945f9f9c
IPT	TipsyCoin/contracts/IPancake.sol	e751a1187acb72885a3f1e7146f9c1c713e44c698a2f151b481263f903f9c150
ITT	TipsyCoin/contracts/ITokenTimelock.sol	f439c283f1a38c10405221f14c5a43f7ee0871aae0e4d902405f7b3ff872380c
ITV	TipsyCoin/contracts/ITokenVesting.sol	0ca72bd4480f2323322273719761993854 3fe1f928d278c4fbbccd74eaed8aac
ITC	TipsyCoin/contracts/Imports.sol	4b2aea612d407d5d7b5a78c9973852ccaf344b1c7a850cdc6e9f7f5ee01f7e25
TLT	TipsyCoin/contracts/TimeLock.sol	f51faa9ee0ac3d0aef40466d3a7e9be2b811d41cbcb63b4a74ead4dd56d35a81
TCT	TipsyCoin/contracts/TipsyCoin.sol	1cdcd1b72f509dd14159956cc994c40e8637d68332a2caabe3ad9ebe93e2524e
THT	TipsyCoin/contracts/TokenHolder.sol	e4cbb34fbbf3cfdb05ee02647d449730ccd1a9e66d81517a6f4093f7a4bb84f8
TVT	TipsyCoin/contracts/TokenVesting.sol	6ddaea9a3ff015ead3b0b2c0891abbabecc7a6f426cd3e0834f8a4a50fcca4c0

Findings



Critical	**0** (0.00%)
Major	**4** (30.77%)
Medium	**3** (23.08%)
Minor	**1** (7.69%)
Informational	**5** (38.46%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
BBT-01	Potential Sandwich Attacks	Logical Issue	Minor	Resolved
DTC-01	Redundant Modifier "onlyOwner"	Coding Style, Gas Optimization	Informational	Resolved
DTC-02	Repeated Checking `nonReentrant`	Gas Optimization	Informational	Resolved
DTC-03	If One Account `payment`==0, All Accounts Can't Get `payment`	Volatile Code	Medium	Resolved
DTC-04	Centralization Risk in Distribution.sol	**Centralization / Privilege**	**Major**	Resolved
TCC-01	Centralized Control of Contract Upgrade	**Centralization / Privilege**	**Major**	Mitigated
TCT-01	Remove Testing Code	Volatile Code	Informational	Resolved
TCT-02	Incorrect Calculation Of Tax	Logical Issue	Medium	Resolved
TCT-03	Miss Comparing `realAmount` and `reflexAmount`	Logical Issue	Medium	Resolved
TCT-04	Missing Emit `transfer` Event	Logical Issue	Informational	Resolved
TCT-05	Potential Sandwich Attacks	Logical Issue	Informational	Acknowledged
TCT-06	Centralization Risk in TipsyCoin.sol	**Centralization / Privilege**	**Major**	Mitigated

ID	Title	Category	Severity	Status
TVT-01	Centralization Risk in TokenVesting.sol	**Centralization / Privilege**	**Major**	Resolved

BBT-01 | Potential Sandwich Attacks

Category	Severity	Location	Status
Logical Issue	Minor	TipsyCoin/contracts/BuyBack.sol: 131, 144, 157~158	Resolved

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.

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:

- `swapExactTokensForTokens()`
- `addLiquidity()`

Recommendation

We recommend setting reasonable minimum output amounts, instead of 1, based on token prices when calling the aforementioned functions.

Alleviation

[Tipsy Team] Uses human oracle to enforce a recent quote on how much TipsyCoin is expected after swapping in wBNB to preform a Buyback. setQuote is a public (but non-spending) function which allows devs to call this without going through the TimelockController. Once the devs have veried that quote is fair, they may then execute a timelocked BuyBack function. setQuote has a short cool-down, to prevent adversaries from calling setQuote during a sandwich attack. Additionally, a "percentageMinOutput" is added to the BuyBack function to allow for slippage. Fixes BBT-01 by using human oracle to get reasonable PCS quote to reduce the impact of potential sandwich attacks: https://github.com/TipsyDev-Mittens/TipsyCoin/commit/0f5cfcb460c49f23f1ec09562fd1b16e6c47c831 . Further, Tipsy team will work to implement TWAP for TipsyCoin (after release), so that reducing the risk of sandwich attacks is less labour intensive in future.

DTC-01 | Redundant Modifier "onlyOwner"

Category	Severity	Location	Status
Coding Style, Gas Optimization	Informational	TipsyCoin/contracts/Distribution.sol: 80	Resolved

Description

The function `_addPayee` is `internal`, which is called by `addNewPayee` and `initialize`. There is no need to add `onlyOwner` for `internal` function.

Recommendation

We recommend removing the `onlyOwner` for `_addPayee`.

Alleviation

Fixed in commit hash `0cefd6f40f6a7bbf4b6eef13c79b308d362b6b9d`.

DTC-02 | Repeated Checking `nonReentrant`

Category	Severity	Location	Status
Gas Optimization	Informational	TipsyCoin/contracts/Distribution.sol: 49	Resolved

Description

The function `_release` is `internal`, which is called by the function `release_all` in a loop, so `nonReentrant` runs repeatedly.

Recommendation

We recommend moving `nonReentrant` to external function `release_all`.

Alleviation

Removed `nonReentrant` from `_release` in commit hash `ce613248b310fc733eaf4428709544cf5a871cf1`.
Fixed in commit `4a594ccda9cfa68fa67240ad33a03caf523eea87`.

DTC-03 | If One Account `payment` ==0, All Accounts Can't Get `payment`

Category	Severity	Location	Status
Volatile Code	Medium	TipsyCoin/contracts/Distribution.sol: 58~61	Resolved

Description

In the function `_release`, it uses `require` statement to ensure `payment != 0` or it reverts the `release_all` function even though the `payment` is not zero for other accounts.

Recommendation

We recommend using `if` condition instead of `require` statement.

```
    if (payment != 0) {
        _tokenReleased[account] = _tokenReleased[account] + payment;
        _totalTokenReleased = _totalTokenReleased + payment;

        IERC20Upgradeable(paymentToken).safeTransfer(account, payment);
        emit PaymentReleased(account, payment);
    }
```

Alleviation

Fixed in commit hash `59a14baea4733efe302edfca858c2319c641423e`.

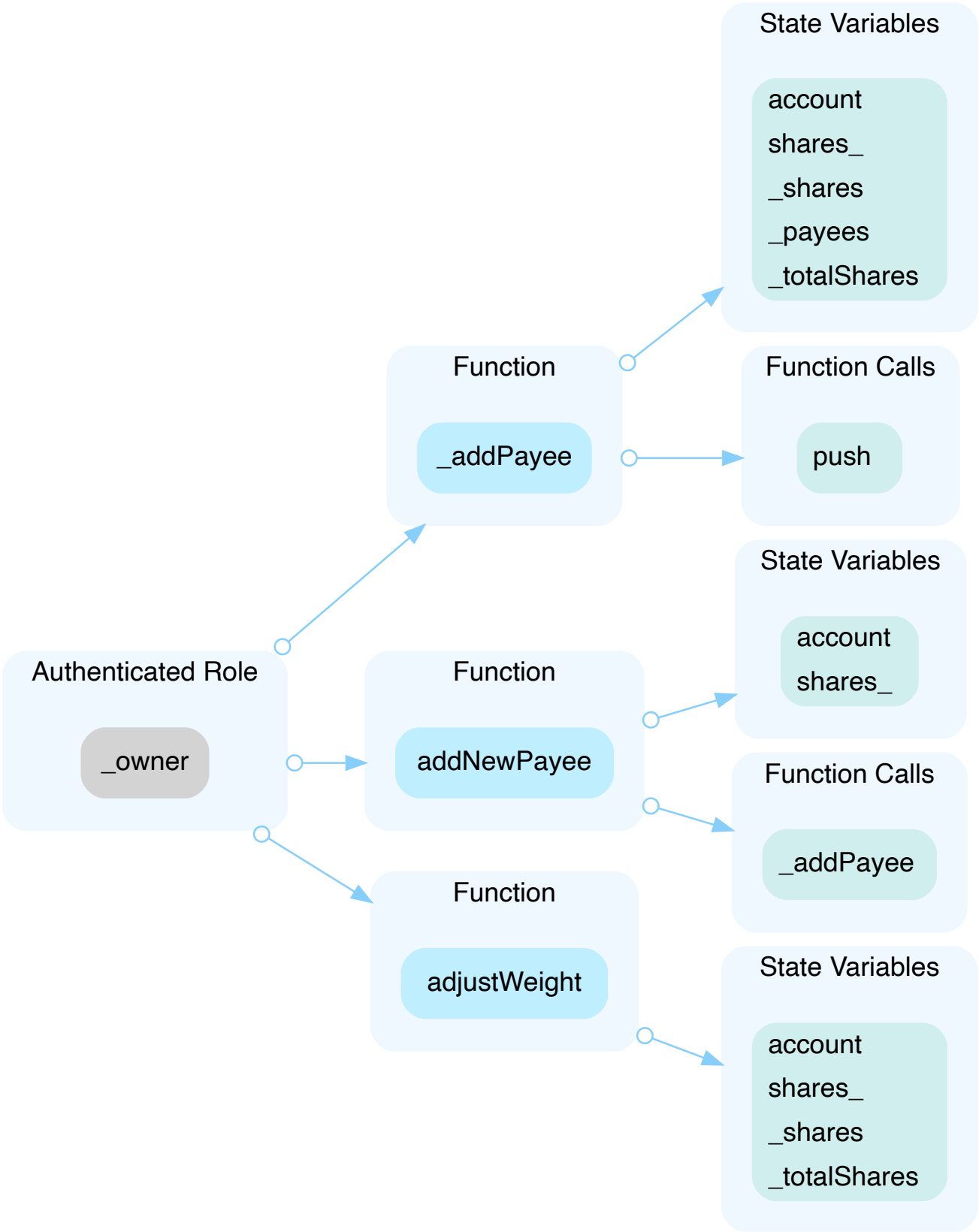
DTC-04 | Centralization Risk In Distribution.sol

Category	Severity	Location	Status
Centralization / Privilege	**Major**	TipsyCoin/contracts/Distribution.sol: 80~94, 96~98, 100~117	Resolved

Description

In the contract, `Distribution`, the role, `_owner`, has authority over the functions shown in the diagram below.

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



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., multisignature 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

Removed `onlyOwner` modifiers in commit 'dd4f26fc2c8a93b2446782e660c745b06d1ff8fb'.

TCC-01 | Centralized Control Of Contract Upgrade

Category	Severity	Location	Status
Centralization / Privilege	Major	TipsyCoin/contracts/BuyBack.sol: 83 TipsyCoin/contracts/Distribution.sol: 11 TipsyCoin/contracts/TimeLock.sol: 18 TipsyCoin/contracts/TipsyCoin.sol: 83 TipsyCoin/contracts/TokenHolder.sol: 8 TipsyCoin/contracts/TokenVesting.sol: 18	Mitigated

Description

Contracts `TipsyBuyBack`, `Distribution`, `TokenTimelock`, `TipsyCoin`, `BasicTokenContract` and `TokenVesting` are upgradeable contract, the owner can upgrade the contract without the community's commitment. If an attacker compromises the account, he can change the implementation of the contract and drain tokens from the contract.

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.

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

[Tipsy Team]: TipsyCoin has added a standard implementation of OpenZepplin's TimelockController to the project in order to mitigate the centralisation issues discussed in Certik's audit of TipsyCoin: github.com/TipsyDev-Mittens/TipsyCoin/commit/ec1f9f850f738663a51deb03faaa3928c22b8bc .

Two TimelockControllers will be used for the deployment (one for ProxyAdmin, one for Owner), both with a 48-hour delay, for any contracts with centralisation concerns. The Proposers() for these TimelockController's will be the 3/5 Gnosis Multisig for ProxyAdmin, and the 2/3 Gnosis Multisig for the Owner().

The Executors() for these TimelockController's will be the members of those respective Multisigs, plus the TipsyCoin Deployment wallet.

This info, plus additional commentary and illustration can be found in the updated README.md on TipsyCoin's Github: https://github.com/TipsyDev-Mittens/TipsyCoin#readme

Further, TipsyCoin has set up a Medium page to ensure contracts addresses, and details (including the timelock) will be published to meet CertiK's mitigation suggestions: medium.com/@tipsycoin

Tipsy team will also leave the TipsyCoin Github public, and publish contracts addresses, and details in the README.md via this platform, too.

Finally, Tipsy team has begun the setup of a DAO / voting module using Snapshot to satisfy CertiK's long-term mitigation suggestions – snapshot.org/#/tispycoin.eth/ .

TCT-01 | Remove Testing Code

Category	Severity	Location	Status
Volatile Code	Informational	TipsyCoin/contracts/TipsyCoin.sol: 338	Resolved

Description

There are some testing code blocks in the audit contracts. Because the file hash is in the audit report, it's better to remove all unnecessary code from contract.

Recommendation

We recommend removing testing code.

Alleviation

Fixed in commit hash `fb78b31335969a84bd244f07fb7a6267382eb9ab`.

TCT-02 | Incorrect Calculation Of Tax

Category	Severity	Location	Status
Logical Issue	Medium	TipsyCoin/contracts/TipsyCoin.sol: 524~527	Resolved

Description

According to the comments, the trade tax is 10% of trade amount. `_feeTotal` equals `buybackFundAmount + marketingCommunityAmount + reflexiveAmount`, so `amount * buybackFundAmount / _feeTotal + amount * marketingCommunityAmount / _feeTotal +amount * reflexiveAmount / _feeTotal` is 100%, not 10%.

Recommendation

We recommend correcting the calculation like below.

```
1       uint _amountBuyBack = amount * buybackFundAmount / _feeTotal/10;
2       uint _amountMarketing = amount * marketingCommunityAmount / _feeTotal/10;
3       uint _amountReflexive = amount * reflexiveAmount / _feeTotal/10;
4
```

Alleviation

Fixed in commit hash '61151ffd26825fb8c788bf6312d1ca116bfdc04a'.

TCT-03 | Miss Comparing `realAmount` And `reflexAmount`

Category	Severity	Location	Status
Logical Issue	Medium	TipsyCoin/contracts/TipsyCoin.sol: 547~551, 567, 586~590	Resolved

Description

In the function `transferFrom`, `currentAllowance` is `realAmount`, but `amount` is `reflexAmount`, comparing these two types of value is incorrect.

Recommendation

We recommend using the same type to compare and calculate.

Alleviation

Fixed in commit hash `d1b6a30922fbff591735981bcbe215e40b60da25`.

TCT-04 | Missing Emit `transfer` Event

Category	Severity	Location	Status
Logical Issue	Informational	TipsyCoin/contracts/TipsyCoin.sol: 517	Resolved

Description

In the function `transferFrom`, there is no `transfer` event emitted.

Recommendation

We recommend adding `transfer` event.

Alleviation

Fixed in commit hash `5f0e6b977f96dcfe4368947b997f727f0243dfc4`.

TCT-05 | Potential Sandwich Attacks

Category	Severity	Location	Status
Logical Issue	Informational	TipsyCoin/contracts/TipsyCoin.sol: 494~496	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.

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:

- `pancakeV2Router.swapExactTokensForTokens()`

Recommendation

We recommend setting reasonable minimum output amounts, instead of 1, based on token prices when calling the aforementioned functions.

Alleviation

[Tipsy Team] Tipsy team will work on implementing a TWAP oracle for TipsyCoin after release, and then upgrade the TipsyCoin contract to use this oracle in order to further mitigate potential sandwich attacks.

Additionally, an adversary attempting to sandwich the taxed portion of a TipsyCoin transaction would themselves be subject to the 10% tax. The team's opinion is that an attempted attack like this is very unlikely to be protable.

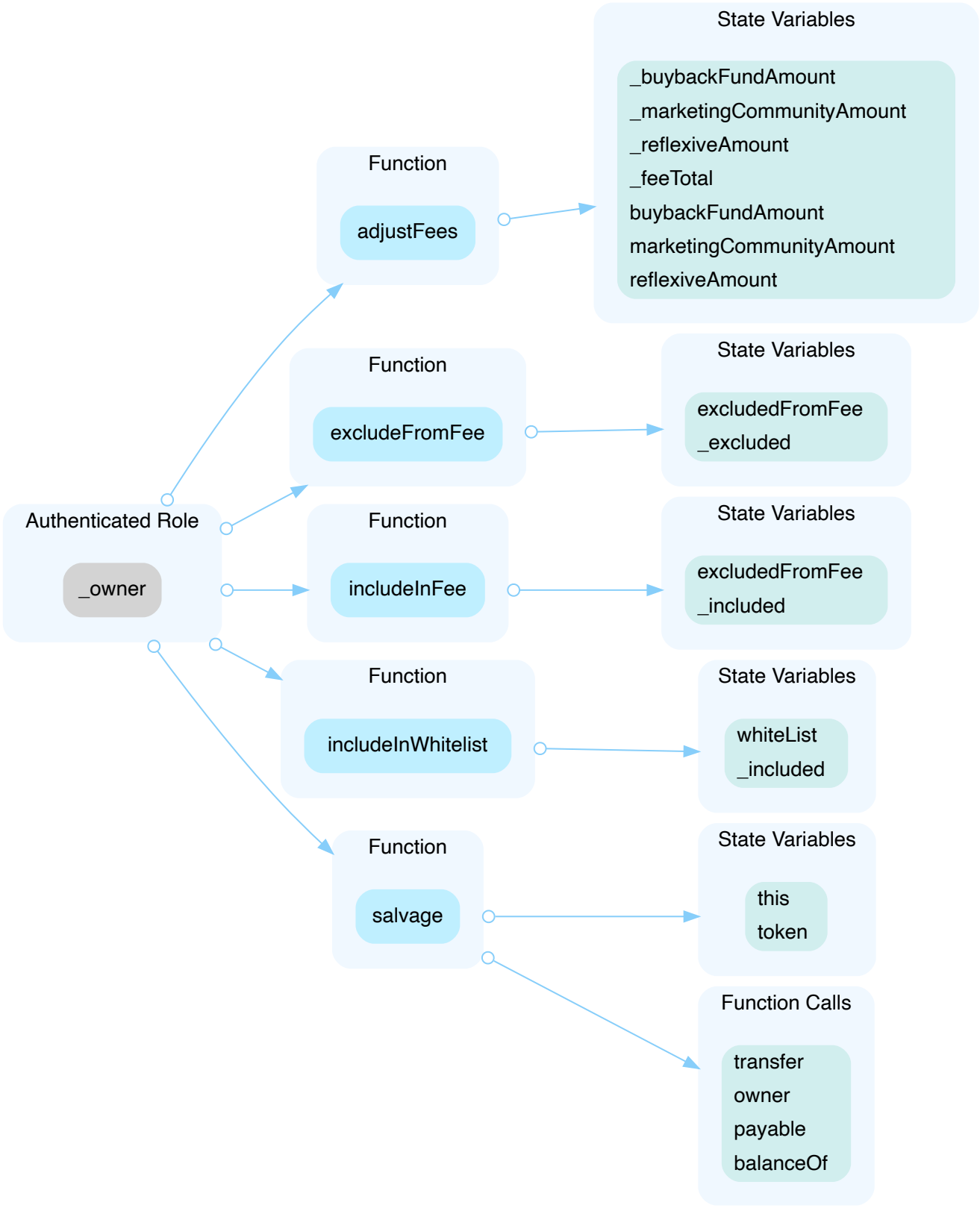
TCT-06 | Centralization Risk In TipsyCoin.sol

Category	Severity	Location	Status
Centralization / Privilege	**Major**	TipsyCoin/contracts/TipsyCoin.sol: 369~376, 383~387, 394~398, 406~410, 427~432	Mitigated

Description

In the contract, `TipsyCoin`, the role, `_owner`, has authority over the functions shown in the diagram below.

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



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., multisignature 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

[Tipsy Team] This fix adds a standard implementation of OpenZepplin's TimelockController to the project in order to resolve the centralisation issues discussed in Certik's audit of TipsyCoin - github.com/TipsyDev-Mittens/TipsyCoin/commit/ec1f9f850f738663a51deb03faaa3928c22b8bc . Two TimelockControllers will be used for the deployment (one for ProxyAdmin, one for Owner), both with a 48 hour delay. The Proposers for these TimelockController's will be The 3/5 Gnosis Multisig for ProxyAdmin, and the 2/3 Gnosis Multisig for the Owner. The Executors for these TimelockControllers will be the members of those respective Multisigs, plus the TipsyCoin Deployment wallet. This info, plus additional commentary and illustration can be found in the updated README.md on TipsyCoin's Github.

Further, TipsyCoin has set up a Medium page to ensure contracts addresses, and details (including the timelock) will be published to meet CertiK's mitigation suggestions: medium.com/@tipsycoin

Tipsy team will also leave the TipsyCoin Github public, and publish contracts addresses, and details in the README.md via this platform, too.

Finally, Tipsy team has begun the setup of a DAO / voting module using Snapshot to satisfy CertiK's long-term mitigation suggestions – snapshot.org/#/tispycoin.eth/.

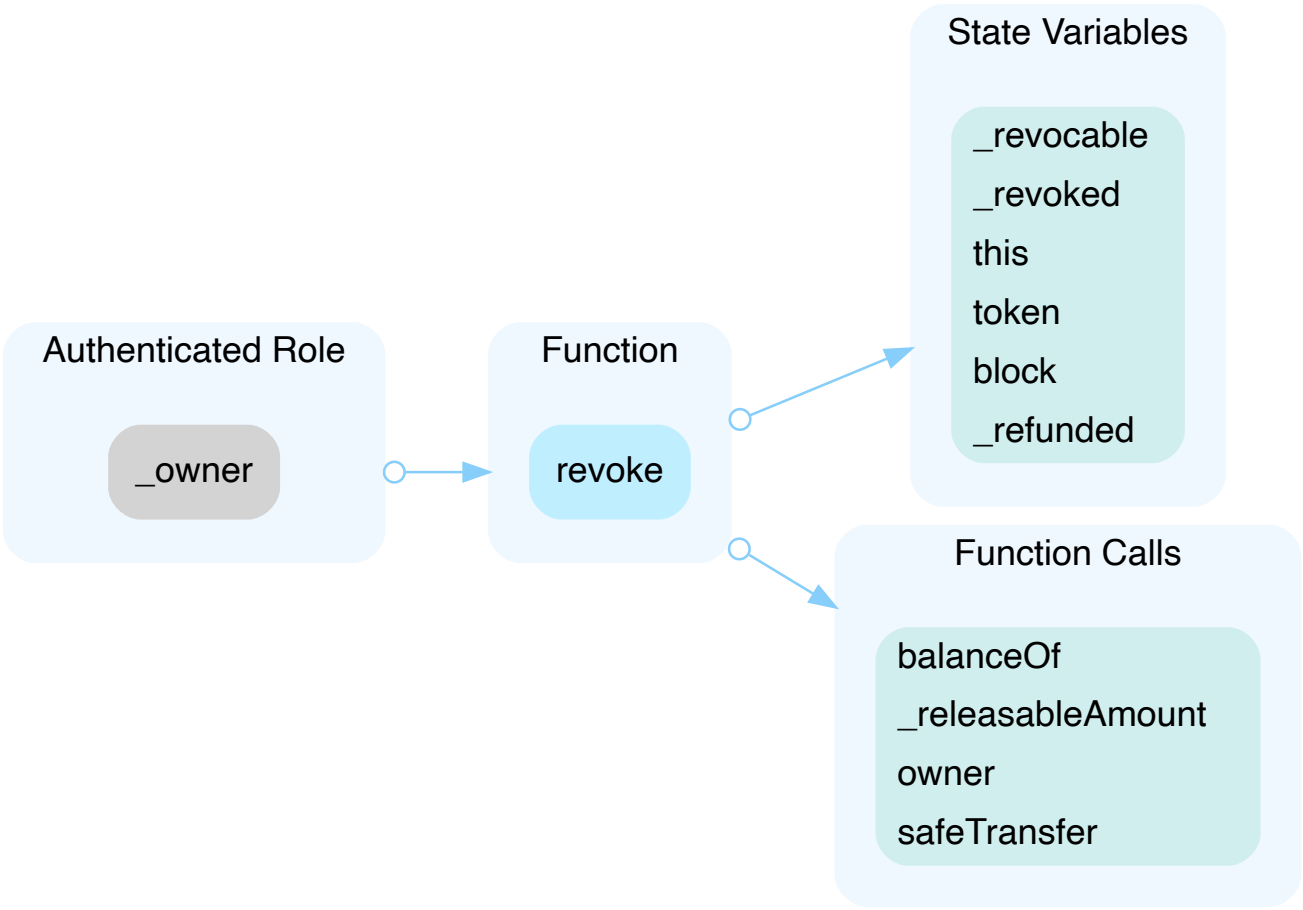
TVT-01 | Centralization Risk In TokenVesting.sol

Category	Severity	Location	Status
Centralization / Privilege	Major	TipsyCoin/contracts/TokenVesting.sol: 121~137	Resolved

Description

In the contract, `TokenVesting`, the role, `_owner`, has authority over the functions shown in the diagram below.

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



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., multisignature 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

Fixed by removing the function `revoke` in commit '45511956bd5b7db53bd6484627e57504efc7160d'.

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.

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.