CERTIK

Security Assessment

Dank Protocol III

Oct 8th, 2021

Table of Contents

Summary

This report has been prepared for Dank digital Co., Ltd to discover issues and vulnerabilities in the source code of the Dank Protocol III 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	Dank Protocol III
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/Dank-Protocol/Decentralized-Bank/tree/master/kovan
Commit	1500b8b1e49f4a369527f70171eab6d07f2862cc

Audit Summary

Delivery Date	Oct 08, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

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

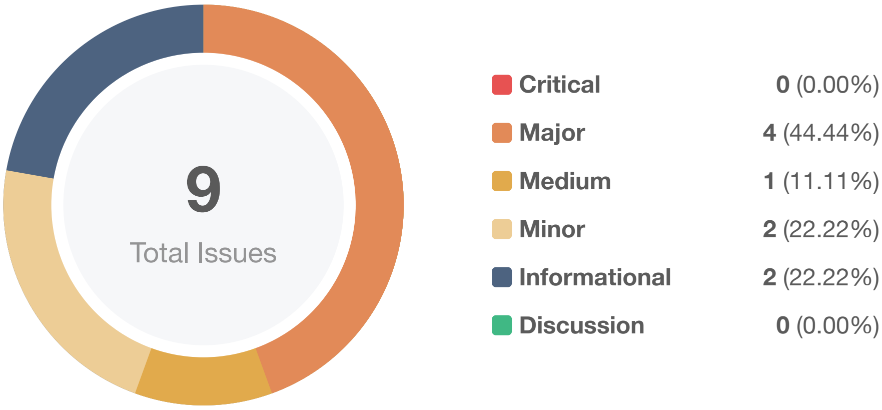
Audit Scope

ID	File	SHA256 Checksum
DGD	Governance/Dank.sol	d5fcbff523875811c468c922766e498de3c203bcf9e00f70d6d6081173097f65
JRM	dETH/JumpRateModel.sol	c75cc1fcb7d915788b7c2342cd09eb49d5c89329052c9e8487ce80a57f31ae2c
AVI	proxy/AggregatorValidatorInterface.sol	28505af2acdb398492d4649ec92655b217a7e825b722bedd67086edca2a798a7
COD	proxy/ConfirmedOwner.sol	9e08f82d045781687f42c90f78c57fef423075fae5e3d7cd33130b01dc123832
MAV	proxy/MockAggregatorValidator.sol	ecaaa49246d1da838bd89f73489e3dfd021b88770b4b18ade2d5c398bb908bb1
TAV	proxy/TypeAndVersionInterface.sol	4e10d008a9aefd8c1fad33a43580684baefd9353f14a76d366192ed7f5c5de08
VPD	proxy/ValidatorProxy.sol	aa2ab1b20655e2569f02b1a6e47a923926680e1a8e8e103795bf0ecbce554107

Financial Models

Financial models of blockchain protocols need to be resilient to attacks. It needs to pass simulations and verifications to guarantee the security of the overall protocol.

Findings



ID	Title	Category	Severity	Status
Dank Protocol-01	Missing input validation	Volatile Code	● Minor	Resolved
Dank Protocol-02	Proper usage of “public” and “external” type	Coding Style	● Informational	Acknowledged
Dank Protocol-03	Incorrect naming convention utilization	Coding Style	● Informational	Acknowledged
DGD-01	Centralization risk	**Centralization / Privilege**	● **Major**	Partially Resolved
DGD-02	Does not move delegates while mint token	**Centralization / Privilege**	● **Major**	Partially Resolved
DGD-03	Potential subtraction overflow	Logical Issue	● Medium	Resolved
DGD-04	Centralization risk	**Centralization / Privilege**	● **Major**	Partially Resolved
VPD-01	Third party dependencies	Volatile Code	● Minor	Acknowledged
VPD-02	Centralization risk	**Centralization / Privilege**	● **Major**	Acknowledged

Dank Protocol-01 | Missing input validation

Category	Severity	Location	Status
Volatile Code	● Minor	Global	✓ Resolved

Description

The given input is missing the check for the non-zero address.

For example,

- contract `Dank`: `danktrollerAddr` in function `setDanktrollerAddress()`

Recommendation

We advise adding the check for the passed-in values to prevent unexpected error.

Alleviation

The team heeded our advice and removed this function in commit `bb8dfed1cd7b0b204e3bd0d0bab53315a3b385e7`.

Dank Protocol-02 | Proper usage of “public” and “external” type

Category	Severity	Location	Status
Coding Style	● Informational	Global	ⓘ Acknowledged

Description

“public” functions that are never called by the contract should be declared “external”. When the inputs are arrays, “external” functions are more efficient than “public” functions.

Examples:

Functions like :

- contract `Dank`: `delegate()`, `delegateBySig()`, `getPriorVotes()`, `owner()`
- contract `JumpRateModel`: `getSupplyRate()`
- contract `ConfirmedOwner`: `owner()`

Recommendation

We recommend using the “external” attribute for functions never called from the contract.

Alleviation

The team acknowledged this issue and they will leave it as it is for now.

Dank Protocol-03 | Incorrect naming convention utilization

Category	Severity	Location	Status
Coding Style	● Informational	Global	ⓘ Acknowledged

Description

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

Constants should be named with all capital letters with underscores separating words UPPER_CASE_WITH_UNDERSCORES

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

Examples:

Constants like :

- contract `Dank`: `maxTotalSupply`, `perBlockMint`
- contract `JumpRateModel`: `isInterestRateModel`, `blocksPerYear`
- contract `ConfirmedOwner`: `s_owner`, `s_pendingOwner`
- contract `ValidatorProxy`: `s_currentAggregator`, `s_proposedAggregator`, `s_currentValidator`, `s_proposedValidator`

Recommendation

The recommendations outlined here are intended to improve the readability, and thus they are not rules, but rather guidelines to try and help convey the most information through the names of things.

Alleviation

The team acknowledged this issue and they will leave it as it is for now.

DGD-01 | Centralization risk

Category	Severity	Location	Status
Centralization / Privilege	● **Major**	Governance/Dank.sol: 189, 274, 299	Partially Resolved

Description

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

- `setDanktrollerAddress()`: change the `danktrollerAddr` to any addresses.
- `_delegate()/_transferTokens()`: mint tokens to the `danktrollerAddr` and the `ownerAddr`.
- `transferOwnership()/acceptOwnership()`: change the `owneraddr` to any addresses.

Any compromise to the `owner` account may allow the hacker to take advantage of this and users' assets may suffer loss.

Recommendation

We advise the client to carefully manage the `owner` account'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 are some feasible suggestions that would also mitigate the potential risk at the different levels in terms 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

The team heeded our advice and removed the function `setDanktrollerAddress()` and the logic of mint tokens in the functions `_delegate()/_transferTokens()` in commit `bb8dfed1cd7b0b204e3bd0d0bab53315a3b385e7`.

They will renounce ownership to a multi-sig governing procedure on their own timeframe.

DGD-02 | Does not move delegates while mint token

Category	Severity	Location	Status
Centralization / Privilege	● Major	Governance/Dank.sol: 285, 311	Partially Resolved

Description

In essence, DANK Token lets token holders delegate their voting power to another entity.

According to the following codes, new DANK tokens will be minted and distributed to the `danktrollerAddr` and the `ownerAddr` according to the block number.

```
 278          uint diffBlock = block.number.sub(currBlock);
 279          if (diffBlock > 0) {
 280              uint diffAmount = diffBlock.mul(perBlockMint);
 281
 282              uint maltAmount = diffAmount.mul(5).div(10);
 283              uint balanceDanktroller =
balanceOfHold(getDanktrollerAddress()).add(maltAmount);
 284              balances[getDanktrollerAddress()] = safe96(balanceDanktroller,
"Dank:_delegate: amount exceeds 96 bits");
 285              balances[ownerAddr] = safe96(balanceOfHold(ownerAddr).add(maltAmount),
"Dank:_delegate: amount exceeds 96 bits");
 286
 287              currBlock = block.number;
 288          }
```

```
 304          uint diffBlock = block.number.sub(currBlock);
 305          if (diffBlock > 0) {
 306              uint diffAmount = diffBlock.mul(perBlockMint);
 307
 308              uint maltAmount = diffAmount.mul(5).div(10);
 309              uint balanceDanktroller =
balanceOfHold(getDanktrollerAddress()).add(maltAmount);
 310              balances[getDanktrollerAddress()] = safe96(balanceDanktroller,
"Dank:_transferTokens: amount exceeds 96 bits");
 311              balances[ownerAddr] = safe96(balanceOfHold(ownerAddr).add(maltAmount),
"Dank:_transferTokens: amount exceeds 96 bits");
 312
 313              currBlock = block.number;
 314          }
```

However, functions `_delegate()` and `_transferTokens()` don't call `_moveDelegates()` for these minted tokens.

Recommendation

We recommend adding call of `_moveDelegates()` in the functions `_delegate()` and `_transferTokens()` for these minted tokens.

Alleviation

The team heeded our advice and removed this logic in commit `bb8dfed1cd7b0b204e3bd0d0bab53315a3b385e7`, and removed the function `issue()` in commit `748b4369d7ae53e05178cc9950a1d21c9c80b218`.

DGD-03 | Potential subtraction overflow

Category	Severity	Location	Status
Logical Issue	● Medium	Governance/Dank.sol: 328	✓ Resolved

Description

Refer to DGD-02, the new-minted tokens are added to the `ownerAddr`, but `_moveDelegates()` is not called.

According to the following codes, the parameter `srcRep` is `delegates[ownerAddr]` when the `ownerAddr` call the functions `_delegate()` and `_transferTokens()`.

```
    function _delegate(address delegator, address delegatee) internal {
        ...
        _moveDelegates(currentDelegate, delegatee, delegatorBalance);
    }
```

```
    function _transferTokens(address src, address dst, uint96 amount) internal {
        ...
        _moveDelegates(delegates[src], delegates[dst], amount);
    }
```

```
323      function _moveDelegates(address srcRep, address dstRep, uint96 amount) internal {
324          if (srcRep != dstRep && amount > 0) {
325              if (srcRep != address(0)) {
326                  uint32 srcRepNum = numCheckpoints[srcRep];
327                  uint96 srcRepOld = srcRepNum > 0 ? checkpoints[srcRep][srcRepNum - 1].votes : 0;
328                  uint96 srcRepNew = sub96(srcRepOld, amount, "Dank::_moveVotes: vote amount underflows");
329                  _writeCheckpoint(srcRep, srcRepNum, srcRepOld, srcRepNew);
330              }
331
332              if (dstRep != address(0)) {
333                  uint32 dstRepNum = numCheckpoints[dstRep];
334                  uint96 dstRepOld = dstRepNum > 0 ? checkpoints[dstRep][dstRepNum - 1].votes : 0;
335                  uint96 dstRepNew = add96(dstRepOld, amount, "Dank::_moveVotes: vote amount overflows");
336                  _writeCheckpoint(dstRep, dstRepNum, dstRepOld, dstRepNew);
337              }
```

```
        }
    }
```

The value of `delegates[ownerAddr]` can be set and the logic of `srcRep != address(0)` will trigger. As a result, the subtraction `sub96(srcRepOld, amount, "Dank::_moveVotes: vote amount underflows")` may overflow and lead the call to fail.

Recommendation

We recommend fixing DGD-02 or stating for `delegates[ownerAddr]`.

Alleviation

The team removed the logic of mint tokens in the functions `_delegate()/_transferTokens()`, and removed the function `issue()` in commit `748b4369d7ae53e05178cc9950a1d21c9c80b218`.

DGD-04 | Centralization risk

Category	Severity	Location	Status
Centralization / Privilege	● **Major**	Governance/Dank.sol	Partially Resolved

Description

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

- `constructor()`: mint 10000000e18 DANK tokens to the `owner` addresses.
- `issue()`: mint any amount tokens to any addresses.

Any compromise to the `owner` account may allow the hacker to take advantage of this and users' assets may suffer loss.

Recommendation

We advise the client to carefully manage the `owner` account'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 are some feasible suggestions that would also mitigate the potential risk at the different levels in terms 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

The team heeded our advice and removed the function `issue()` in commit `748b4369d7ae53e05178cc9950a1d21c9c80b218`.

They will renounce ownership to a multi-sig governing procedure on their own timeframe.

VPD-01 | Third party dependencies

Category	Severity	Location	Status
Volatile Code	● Minor	proxy/ValidatorProxy.sol: 128, 139	ⓘ Acknowledged

Description

The contract is serving as the underlying entity to interact with third party `AggregatorValidator` 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 of the contract `ValidatorProxy` requires interaction with `AggregatorValidator`, etc. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

The team acknowledged this issue and they will leave it as it is for now.

VPD-02 | Centralization risk

Category	Severity	Location	Status
Centralization / Privilege	● **Major**	proxy/ValidatorProxy.sol: 159, 176, 223, 240	ⓘ Acknowledged

Description

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

- proposeNewAggregator()/upgradeAggregator(): change the `s_currentAggregator.target` to any addresses,
- proposeNewValidator()/upgradeValidator(): change the `s_currentValidator.target` to any addresses,

Any compromise to the `owner` account may allow the hacker to take advantage of this and users' assets may suffer loss.

Recommendation

We advise the client to carefully manage the `owner` account'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

The team acknowledged this issue and they stated the following:

They will renounce ownership to a multi-sig governing procedure on their own timeframe.

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.

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.