blockdag.network - Vesting

Security Assessment

CertiK Assessed on May 1st, 2025

CertiK Assessed on May 1st, 2025

blockdag.network - Vesting

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES	ECOSYSTEM	METHODS
Vesting	EVM Compatible	Formal Verification, Manual Review, Static Analysis

LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 05/01/2025	N/A

CODEBASE

Private Codebase

View All in Codebase Page

Highlighted Centralization Risks

- Contract upgradeability
- Withdraws can be disabled

Vulnerability Summary

7	6	1	0	0	0
Total Findings	Resolved	Multi-Sig & Timelock	Partially Resolved	Acknowledged	Declined

1	Centralization	1 Multi-Sig & Timelock	Centralization findings highlight privileged roles & functions and their capabilities, or instances where the project takes custody of users' assets.
0	Critical		Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
2	Major	2 Resolved	Major risks may include logical errors that, under specific circumstances, could result in fund losses or loss of project control.
2	Medium	2 Resolved	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
2	Minor	2 Resolved	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.

0 Informational	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | BLOCKDAG.NETWORK - VESTING

CODEBASE | BLOCKDAG.NETWORK - VESTING

Repository

Private Codebase

AUDIT SCOPE | BLOCKDAG.NETWORK - VESTING

1 file audited ● 1 file with Resolved findings

ID	File	SHA256 Checksum
● TVC	TreasuryVesting.sol	d84d397ad28c1c3aaff9667d012f82fdbf3b56b2a2183963bd55574115e40aa8

APPROACH & METHODS | BLOCKDAG.NETWORK - VESTING

This report has been prepared for blockdag.network to discover issues and vulnerabilities in the source code of the blockdag.network - Vesting project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Formal Verification, Manual Review, and Static Analysis techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Testing the smart contracts against both common and uncommon attack vectors;
- 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.

FINDINGS | BLOCKDAG.NETWORK - VESTING

7	0	1	2	2	2	0
Total Findings	Critical	Centralization	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for blockdag.network - Vesting. Through this audit, we have uncovered 7 issues ranging from different severity levels. Utilizing the techniques of Formal Verification, Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
TVC-02	**Centralization Risks In TreasuryVesting.Sol**	**Centralization**	**Centralization**	**2/3 Multi-Sig, 24h Timelock**
TVC-03	**`releaseTokens()` Can Only Be Called By The Token Owner**	**Centralization**	**Major**	**Resolved**
TVV-03	`Timelock` Is Misused	Volatile Code	Major	Resolved
TVC-04	Inconsistent Checks During `AddCategory`	Volatile Code	Medium	Resolved
TVC-05	`allocateTokens()` Allows Allocating More Than `vesting.totalAmount`	Volatile Code	Medium	Resolved
TVC-06	`initialize()` Is Unprotected	Logical Issue	Minor	Resolved
TVC-07	Wrong Argument Of `OperationCancelled` Event	Inconsistency	Minor	Resolved

TVC-02 | CENTRALIZATION RISKS IN TREASURYVESTING.SOL

Category	Severity	Location	Status
Centralization	Centralization	TreasuryVesting.sol (base): 161, 212, 305, 397, 412, 430, 443, 461	2/3 Multi-Sig, 24h Timelock

Description

In the `TreasuryVesting` contract, the `ADMIN_ROLE` has control over the following functions:

- `addCategory()`
- `releaseTokens()`
- `batchRelease()`
- `pause()` / `unpause()`
- Manage all roles
- Upgrade the contract logic

The `OPERATOR_ROLE` is authorized to execute the following functions:

- `allocateTokens()`
- `batchAllocate()`

If the `ADMIN_ROLE` is compromised, an attacker could exploit this authority to pause or block the token release process or assign additional operators.

However, since all `bdagToken` balances are kept in the `ADMIN_ROLE` account and not within the contract, any compromise would affect token balances rather than the contract itself.

If the `OPERATOR_ROLE` is compromised, an attacker could leverage this authority to allocate any amount of `bdagToken` to any account.

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

[CertiK, 05/01/2025]: The team has deployed the TreasuryVesting contract at 0xAfE546948CD9a49A33676F2b97ae10C709021811 of Primordial Testnet BDAG Network. It is available via the ERC1967 Proxy contract at 0x950b35E066e9Ff99b17E94cdB6f7CBc1De6d5f96.

The Timelock with 24-hour minimal delay was deployed at 0x1e5395ceab99D0b56DF036F2f4F0ACA36a69946C and assigned as `timelockContract` of the TreasuryVesting.

In addition, the Multi-Party Computation (MPC) multisignature was established via FORDEFI service managing the wallet 0x02249f8B88A1E4Fe211676e55311e1c0DDd8F748. It was assigned as `ADMIN_ROLE` of the TreasuryVesting at transaction 0xd2e2276ab0704b960a4b6ff3c7214b42982f9c63992d5882fa6972c85bc73747.

TVC-03 `releaseTokens()` CAN ONLY BE CALLED BY THE TOKEN OWNER

Category	Severity	Location	Status
Centralization	**Major**	**TreasuryVesting.sol (base): 353**	**Resolved**

Description

The `releaseTokens()` and `batchRelease()` functions are intended to be called by the account that holds the `bdagToken`. This approach undermines the concept of vesting, which is designed to allow users to access their tokens independently, without relying on the actions of other parties.

Recommendation

We recommend depositing of vested tokens to the contract balance to ensure the `releaseTokens()` will always succeed.

Alleviation

[Blockdag Network, 02/25/2025]: bdag is our native token , it's not listed anymore so how can our client have it in their wallet in order to trigger call in vesting contract , no bdag no fees

[CertiK, 02/25/2025]: The team has reviewed the issue and has chosen not to make changes within the scope of the audit. They clarified that the vested/distributed ERC20 `bdagToken` is intended to be a **native token** of the Blockdag chain. Since users do not possess native tokens at the time of distribution, only a privileged role is authorized to release them.

[blockdag.network, 03/25/2025]: The team heeded the advice and resolved the issue in V11 code by depositing tokens in `allocateTokens()`.

TVV-03 | `Timelock` IS MISUSED

Category	Severity	Location	Status
Volatile Code	Major	TreasuryVestingV11.sol (updateV11): 307	Resolved

Description

1. Implementing of Timelock or Multisig functionality as part of a business logic contract is not recommended.
2. `contract ProjectTimelock` is never used and has the same implementation as `TimelockController`.
3. `TIMELOCK_ADMIN_ROLE` is allowed to grant any desired address `DEFAULT_ADMIN_ROLE` / `ADMIN_ROLE` privileges.
4. The idea of `timelockAddress` is unclear. Any EoA can be assigned as "timelock". It is granted `ADMIN_ROLE` and can schedule/execute all the privileged operations. The same way contract ownership can be transferred to the timelock contract without additional roles.
5. The reason to `scheduleAddCategory()` via timelock is unclear.
6. `executeAddCategory()` and several other functions are duplicated.
7. `executeEmergencyPause()` and other `execute` functions don't check the operation type. As a result, the contract can be paused/unpaused using `operationId` related to `OPERATION_ADD_CATEGORY`.
8. There is no reason to `cleanupExpiredOperation()`.
9. Cancelling of `OPERATION_ADD_CATEGORY` emits the category as `OperationCancelled` `operationType` argument.
10. `BatchAllocationStarted` / `BatchAllocationCompleted` are never used.

Recommendation

We recommend using a separate Timelock/Multisig contract and assign it as owner of TreasuryVesting as described in the document.

Alleviation

[blockdag.network, 04/01/2025]: The team heeded the advice and resolved the issueю

TVC-04 | INCONSISTENT CHECKS DURING `AddCategory`

Category	Severity	Location	Status
Volatile Code	Medium	TreasuryVesting.sol (base): 170	Resolved

Description

1. `scheduleAddCategory()` doesn't check that `totalAmount` is non-zero, however, assumes that.
2. The check `categoryVestings[category].totalAmount == 0` from `scheduleAddCategory()` should be performed in `executeAddCategory()` instead. Because the same category can be added between calls.
3. Category-specific validations should be performed in `scheduleAddCategory()` instead of `executeAddCategory()` to prevent wrong operations to be added.
4. It is not validated that the last `timeSteps` element is equal to `duration`.
5. `start` is not validated to be non-zero.
6. `timeSteps` are not validated to be ascending.
7. It is unclear if `category` can be one of three specific values or any other.

Recommendation

We recommend adding additional checks.

Alleviation

[Blockdag Network, 02/25/2025]: solved in V6

[CertiK, 02/25/2025]: The team heeded the advice and resolved the issue in V6 of the vesting contract `TreasuryVestingV6`, which sha256 checksum is d5a0779fc6eb45d460577987daf033b9b48a98d96bda028d67d82e97ea7b481a

TVC-05 `allocateTokens()` ALLOWS ALLOCATING MORE THAN `vesting.totalAmount`

Category	Severity	Location	Status
Volatile Code	Medium	TreasuryVesting.sol (base): 314	Resolved

Description

```
314
        require(vesting.released + amount <= vesting.totalAmount, "Exceeds category limit");
```

`allocateTokens()` checks if `vesting.totalAmount` is not exceeded. However, the `vesting.released` is only updated in `releaseTokens()`. As a result, an unlimited amount of tokens can be allocated before they become vested.

Recommendation

We recommend decreasing `vesting.totalAmount` when new tokens are allocated. We recommend using of non-zero `start` as a sign of existing `category`.

Alleviation

[Blockdag Network, 02/25/2025]: solved in V6

[CertiK, 02/25/2025]: The team heeded the advice and resolved the issue in V6 of the vesting contract `TreasuryVestingV6`, which sha256 checksum is d5a0779fc6eb45d460577987daf033b9b48a98d96bda028d67d82e97ea7b481a

TVC-06 | `initialize()` IS UNPROTECTED

Category	Severity	Location	Status
Logical Issue	Minor	TreasuryVesting.sol (base): 13	Resolved

Description

The `TreasuryVesting` logic contract does not protect the initializer. An attacker can front-run the `initialize` call and assume ownership of the logic contract. Once in control, the attacker can perform privileged operations, misleading users into believing that they are interacting with the legitimate owner of the upgradeable contract.

Recommendation

We recommend adding

```
/// @custom:oz-upgrades-unsafe-allow constructor
constructor() initializer {}
```

The addition will prevent the function `initialize()` from being called directly in the implementation contract, but the proxy will still be able to `initialize()` its storage variables.

Alleviation

[Blockdag Network, 02/25/2025]: solved in V6

[CertiK, 02/25/2025]: The team heeded the advice and resolved the issue in V6 of the vesting contract `TreasuryVestingV6`, which sha256 checksum is d5a0779fc6eb45d460577987daf033b9b48a98d96bda028d67d82e97ea7b481a

TVC-07 | WRONG ARGUMENT OF `OperationCancelled` EVENT

Category	Severity	Location	Status
Inconsistency	Minor	TreasuryVesting.sol (base): 406	Resolved

Description

```
404         emit OperationCancelled(
405             operationId,
406             keccak256(operation.encodedParams),
407             msg.sender
408         );
```

The second argument of `OperationCancelled` is `operationType`, not `encodedParams`. Also `operation` was already deleted from storage, so `operation.encodedParams` is empty.

Recommendation

We recommend emitting of operation type.

Alleviation

[Blockdag Network, 02/25/2025]: solved in V7

[CertiK, 02/25/2025]: The team heeded the advice and resolved the issue in V7 of the vesting contract `TreasuryVestingV7`, which sha256 checksum is 360444c0365b23ec00f6242a8b26df5307e424ba7d5c9b1d284d46ba6b5dda1f

OPTIMIZATIONS | BLOCKDAG.NETWORK - VESTING

ID	Title	Category	Severity	Status
TVC-01	`OPERATION_UPDATE_SCHEDULE` Is Unused	Code Optimization	Optimization	Resolved

TVC-01 | `OPERATION_UPDATE_SCHEDULE` IS UNUSED

Category	Severity	Location	Status
Code Optimization	Optimization	TreasuryVesting.sol (base): 41, 47, 82	Resolved

Description

`CategoryScheduleUpdated` event and `OPERATION_UPDATE_SCHEDULE` constant are never used.

Recommendation

We recommend removing of unused declarations.

Alleviation

[Blockdag Network, 02/25/2025]: ok see V7

[CertiK, 02/25/2025]: The team heeded the advice and resolved the issue in V7 of the vesting contract `TreasuryVestingV7`, which sha256 checksum is 360444c0365b23ec00f6242a8b26df5307e424ba7d5c9b1d284d46ba6b5dda1f

APPENDIX | BLOCKDAG.NETWORK - VESTING

Finding Categories

Categories	Description
Inconsistency	Inconsistency findings refer to different parts of code that are not consistent or code that does not behave according to its specification.
Volatile Code	Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases and may result in vulnerabilities.
Logical Issue	Logical Issue findings indicate general implementation issues related to the program logic.
Centralization	Centralization findings detail the design choices of designating privileged roles or other centralized controls over the code.

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 | CERTIK

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.

Elevating Your Entire Web3 Journey

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.