CERTIK

Security Assessment

busdchain.com

CertiK Verified on Jan 3rd, 2023

CertiK Verified on Jan 3rd, 2023

busdchain.com

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

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Binance Smart Chain (BSC)	Manual Review, Static Analysis
LANGUAGE	**TIMELINE**	**KEY COMPONENTS**
Solidity	Delivered on 01/03/2023	N/A

CODEBASE

https://bscscan.com/address/0xd55bb498716aae08a761077e1a196fd42ce669af

...View All

Vulnerability Summary

16	0	0	0	16	0	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined	Unresolved

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 Acknowledged	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
2	Medium	2 Acknowledged	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
9	Minor	9 Acknowledged	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.
3	Informational	3 Acknowledged	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 | BUSDCHAIN.COM

CODEBASE | BUSDCHAIN.COM

Repository

https://bscscan.com/address/0xd55bb498716aae08a761077e1a196fd42ce669af

AUDIT SCOPE | BUSDCHAIN.COM

14 files audited ● 5 files with Acknowledged findings ● 9 files without findings

ID	File	SHA256 Checksum
OWN	@openzeppelin/contracts/access/Ownable.sol	dc6ecf2fb375c223c78b1eecb52d9ddf2397a622af29aa39597bf9fa5e800ad4
UBC	@openzeppelin/contracts/proxy/beacon/UpgradeableBeacon.sol	ff4b93b3233de0eef4403bac96698ca7d1874d01e8d1456c225c26a770c9d583
PAC	@openzeppelin/contracts/proxy/transparent/ProxyAdmin.sol	5ff6b0158b0dae1d89d1bcb8e2b37ee80ef0a56e5fdbf9dfadb8e0c85e5c0def
IMP	contracts/import.sol	fbd2dbc1a472e4e58973c7554b906b2fb5012114018ce69bf6f13a0de5b949fa
BUC	BUSDChain.sol	493b71a5d7f9e8648cafbd4a175c521f60a973cafafcfb442702b299be08b9d5
ERP	@openzeppelin/contracts/proxy/ERC1967/ERC1967Proxy.sol	13d890d68e3dba5ffa21db23a4a1cf77e691e3325dea87dd0c77e25e4fb27a85
ERU	@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol	284497022a4580c053a0fd7eb13d6f65f5cd9bcdd6ae542afe88def305c9c31e
BPC	@openzeppelin/contracts/proxy/beacon/BeaconProxy.sol	4f5a55924b1ed6d05bfd0bdee9f0741cd73a638dc6d6124cc840c7e831dcb663
IBC	@openzeppelin/contracts/proxy/beacon/IBeacon.sol	312cb1d7e14511ac958fe4963fb2e4154f42d9939f005d5703def1a1a1f21aa5
TUP	@openzeppelin/contracts/proxy/transparent/TransparentUpgradeableProxy.sol	5789897d5cd1449f6d2a55e61cd99b6fede0379133c8f1a2026a81dd025d1d03
PRX	@openzeppelin/contracts/proxy/Proxy.sol	779ea8a3b6f79496aaf5db95d308a733537f919e543fca914e21f28af3ddd899
ADD	@openzeppelin/contracts/utils/Address.sol	3cd9dd62a5fdc865d8b069f36ee9977c726932b1f6ad9e9bb3acb819dfa6fa59
COT	@openzeppelin/contracts/utils/Context.sol	543c46d0f81fd4e5d9d6a92beef3d2be18badb483b0b4718c819fe3dbbc37587
SSC	@openzeppelin/contracts/utils/StorageSlot.sol	40020f75929aa61b29ad51505de865754bac2d0939050b61ae076255609783af

APPROACH & METHODS | BUSDCHAIN.COM

This report has been prepared for busdchain to discover issues and vulnerabilities in the source code of the busdchain.com project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing 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 | BUSDCHAIN.COM

16	0	2	2	9	3
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for busdchain.com. Through this audit, we have uncovered 16 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
IMP-01	**Centralized Control Of Contract Upgrade**	**Centralization / Privilege**	**Major**	**Acknowledged**
OWN-01	**Centralization Related Risks**	**Centralization / Privilege**	**Major**	**Acknowledged**
BUC-01	Owner Can Register	Volatile Code	Medium	Acknowledged
BUC-02	Incorrect Available Payout	Logical Issue	Medium	Acknowledged
BUC-03	Third Party Dependency	Volatile Code	Minor	Acknowledged
BUC-04	Weak PRNG	Volatile Code	Minor	Acknowledged
BUC-05	Incompatibility With Deflationary Tokens	Logical Issue	Minor	Acknowledged
BUC-06	Unchecked ERC-20 `transfer()` / `transferFrom()` Call	Volatile Code	Minor	Acknowledged
BUC-07	Missing The `whenNotPaused` Modifier	Logical Issue	Minor	Acknowledged
BUC-08	Missing Zero Address Validation	Volatile Code	Minor	Acknowledged

ID	Title	Category	Severity	Status
BUC-09	Potential Loss Of Pool Bonus	Logical Issue	Minor	Acknowledged
BUC-10	'WL', 'WMY' And `dfWallets` Can Not Withdraw Deposit Payout	Logical Issue	Minor	Acknowledged
BUC-11	Potential Whitelist Can Not Withdraw Bonuses As Upline	Logical Issue	Minor	Acknowledged
BUC-12	Hardcode Address	Logical Issue	Informational	Acknowledged
BUC-13	An Extra Loop Is Executed	Logical Issue	Informational	Acknowledged
BUC-14	Discussion : The Use Of The Power Token	Logical Issue	Informational	Acknowledged

IMP-01 | CENTRALIZED CONTROL OF CONTRACT UPGRADE

Category	Severity	Location	Status
Centralization / Privilege	**Major**	**contracts/import.sol (1): 11; BUSDChain.sol (1): 192**	**Acknowledged**

Description

`import` and `BUSDChain` are upgradeable contracts, 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., 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

[Busdchain Team]:

Issue acknowledged. I will fix the issue in the future, which will not be included in this audit engagement.

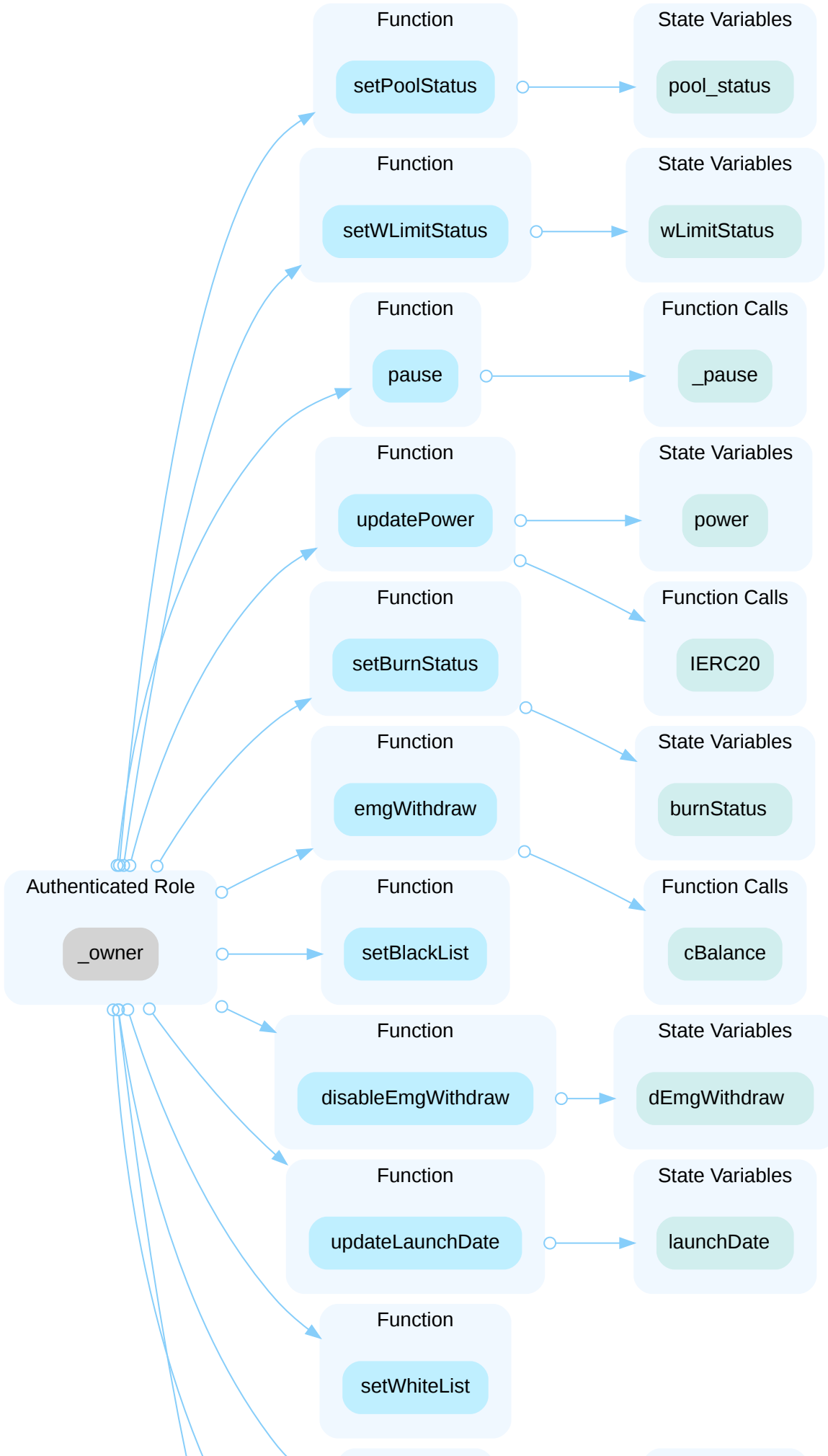
OWN-01 | CENTRALIZATION RELATED RISKS

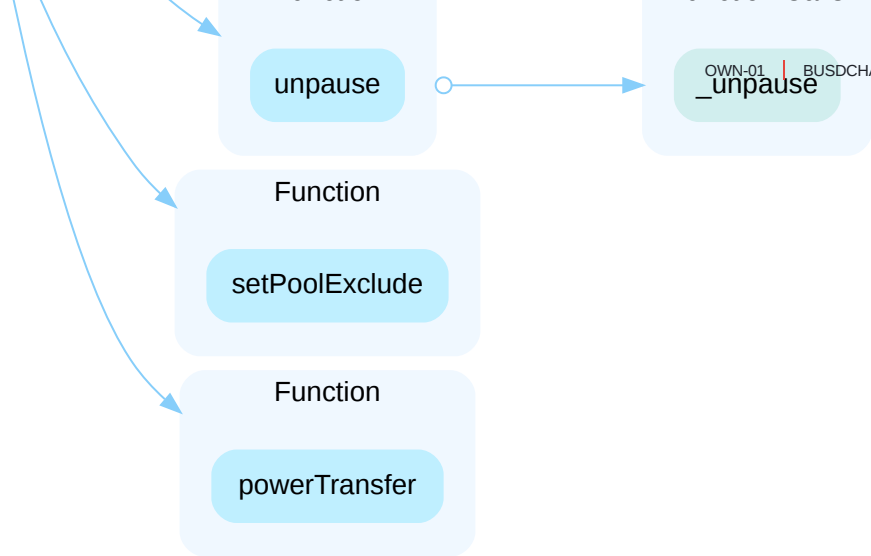
Category	Severity	Location	Status
Centralization / Privilege	**Major**	**@openzeppelin/contracts/access/Ownable.sol (1): 54, 63; @openzeppelin/contracts/proxy/beacon/UpgradeableBeacon.sol (1): 48; @openzeppelin/contracts/proxy/transparent/ProxyAdmin.sol (1): 51, 62, 74; BUSDChain.sol (1): 566, 695, 699, 833, 838, 843, 848, 853, 858, 864, 871, 883, 898, 903**	**Acknowledged**

Description

In the contract `BUSDChain` 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

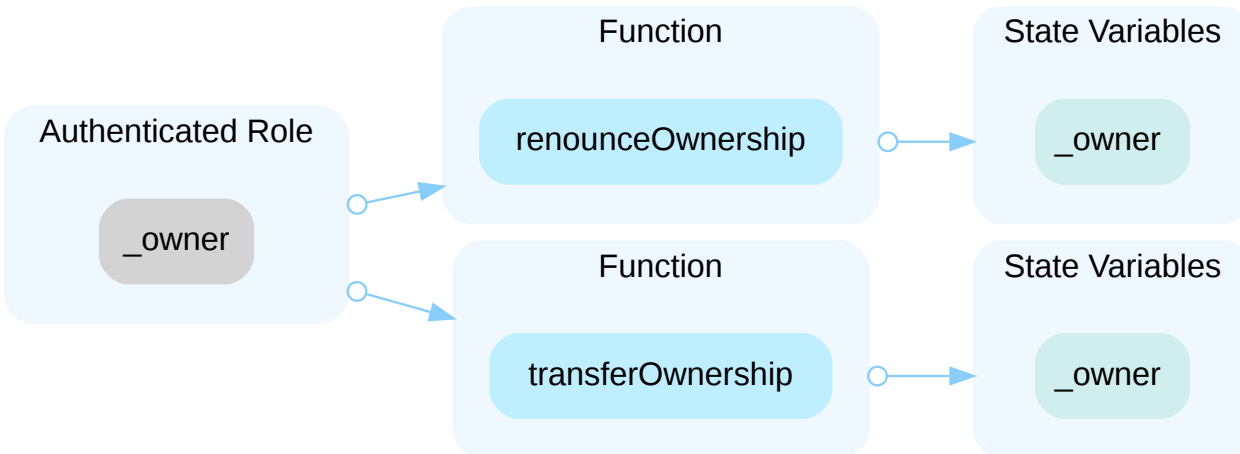
- pause the contract
- unpause the contract and anyone can not withdraw
- whether enable the withdrawal limit check
- whether enable draw pool
- set pool excludes address
- set whitelist
- set blacklist
- whether enable burn power
- transfer power token to any address
- disable emergency withdrawal
- emergency withdraw token to any address
- update the power address
- update launch date


Function
setPoolStatus
State Variables
pool_status
Function
setWLimitStatus
State Variables
wLimitStatus
Function
pause
Function Calls
_pause
Function
updatePower
State Variables
power
Function Calls
IERC20
Function
setBurnStatus
State Variables
burnStatus
Function
emgWithdraw
Function Calls
cBalance
Authenticated Role
_owner
Function
setBlackList
Function
disableEmgWithdraw
State Variables
dEmgWithdraw
Function
updateLaunchDate
State Variables
launchDate
Function
setWhiteList

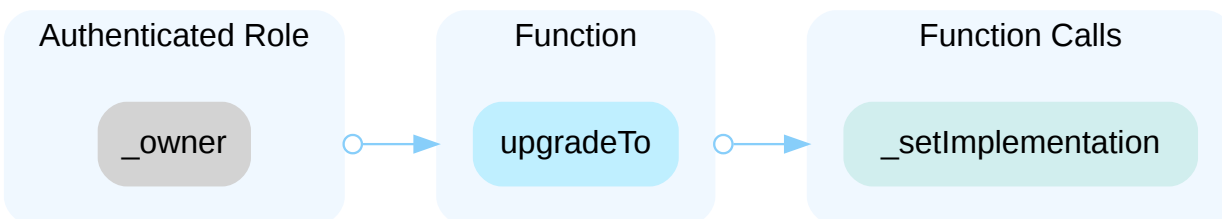


In the contract `BUSDChain`, the blacklist can not withdraw through the function `withdraw()`.

In the contract `Ownable` 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 and renounce and transfer ownership.

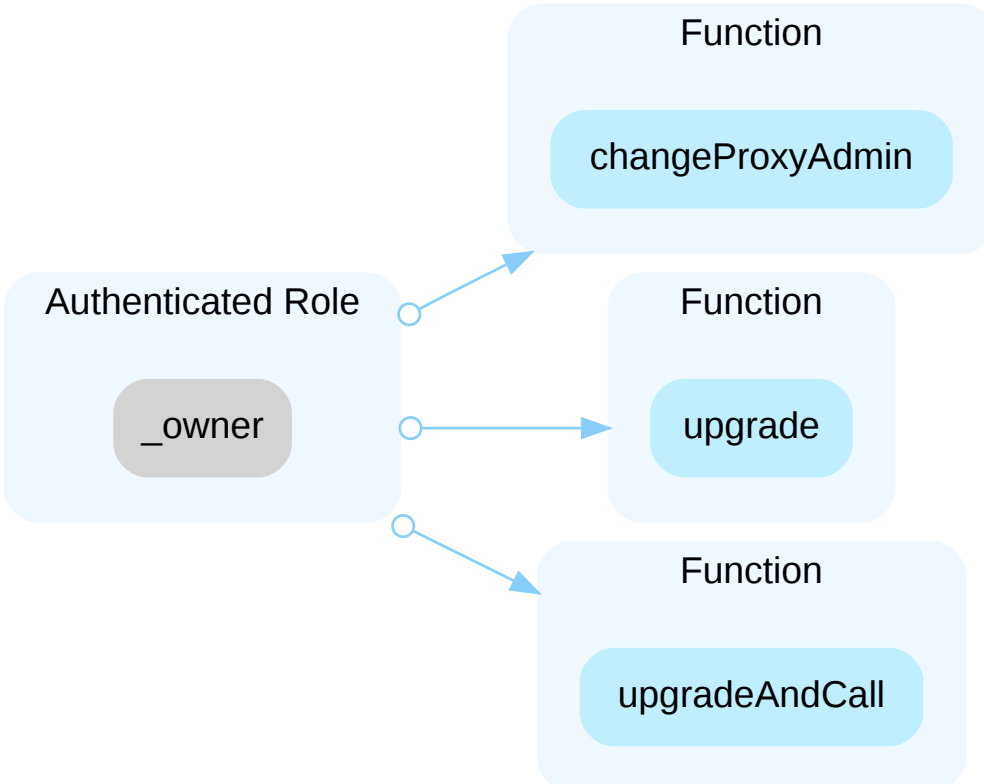


In the contract `UpgradeableBeacon` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and upgrade the beacon to a new implementation.



In the contract `ProxyAdmin` 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 and

- change the admin of `proxy` to `newAdmin`
- upgrade `proxy` to `implementation`
- upgrade `proxy` to `implementation` and call a function on the new implementation



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

[Busdchain Team]:

All these privileges need for project management; as we saw, there aren't critical issues, so we disabled the emergency withdrawal function; here is the transaction:

https://bscscan.com/tx/0x70d2b79d6715fdf50256a24eb7ee3f9c9c7a74892bddc0bb26ccf1deb36aa35a

BUC-01 | OWNER CAN REGISTER

Category	Severity	Location	Status
Volatile Code	Medium	BUSDChain.sol (1): 521	Acknowledged

Description

In the function `initialize()` , the initial upline of the owner is `address(0)` . The function `register()` requires that the upline of the caller is `address(0)` , which means that the owner can call the function to register. If the owner registers the `dfWallets[0]` as his upline, this forms a circle, because the upline of `dfWallets[0]` is the owner.

Recommendation

We advise the client to check if the caller is the owner.

Alleviation

[Busdchain Team]:

The owner knows it and will never register in the project.

BUC-02 | INCORRECT AVAILABLE PAYOUT

Category	Severity	Location	Status
Logical Issue	Medium	BUSDChain.sol (1): 761, 763, 770, 772, 779, 781	Acknowledged

Description

In the function `userAvailable()`, the function `payoutOf()` returns the deposit pending payout and max payout, and the `users[_addr].payouts` is the paid out. If the paid-out is less than the max payout, the direct bonus, pool bonus, and match bonus will be added to the pending payout and returned as a result. However, when processing deposit, direct, and pool payout, the `users[_addr].payouts` is not updated and the `to_payout` is not added. Therefore the returned `to_payout` will be larger than it actually is.

Recommendation

We advise the client to add the `to_payout` when processing deposit, direct, and pool payout.

Alleviation

[Busdchain Team]:

Issue acknowledged. I won't make any changes for the current version.

BUC-03 | THIRD PARTY DEPENDENCY

Category	Severity	Location	Status
Volatile Code	Minor	BUSDChain.sol (1): 195, 197	Acknowledged

Description

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

```
195     IERC20 token;
```

- The contract `BUSDChain` interacts with third party contract with `IERC20` interface via `token`.

```
197     IERC20 power;
```

- The contract `BUSDChain` interacts with third party contract with `IERC20` interface via `power`.

Recommendation

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

Alleviation

[Busdchain Team]:

The project uses BUSD tokens which have a very low risk of third-party issues.

BUC-04 | WEAK PRNG

Category	Severity	Location	Status
Volatile Code	Minor	BUSDChain.sol (1): 529	Acknowledged

Description

Weak PRNG due to a modulo on `block.timestamp`, `block.difficulty`, `msg.sender`, `block.coinbase`, `block.number`, `gasLeft` or `blockhash`. These can be influenced by miners to some extent, so they should be avoided.

```
529                 _upline = dfWallets[random() % 12];
```

Recommendation

Instead of using `block.timestamp`, `block.difficulty`, `msg.sender`, `block.coinbase`, `block.number`, `gasLeft` or `blockhash` as a source of randomness, we recommend using a verifiable source of randomness, such as Chainlink VRF(https://docs.chain.link/docs/get-a-random-number/), for the purpose of random number generation.

Alleviation

[Busdchain Team]:

It is a low-priority random system that will not affect the project and only be used internally.

BUC-05 | INCOMPATIBILITY WITH DEFLATIONARY TOKENS

Category	Severity	Location	Status
Logical Issue	Minor	BUSDChain.sol (1): 409, 436, 553, 563	Acknowledged

Description

When transferring deflationary ERC20 tokens, the input amount may not be equal to the received amount due to the charged transaction fee. For example, if a user sends 100 deflationary tokens (with a 10% transaction fee), only 90 tokens actually arrived to the contract. However, a failure to discount such fees may allow the same user to withdraw 100 tokens from the contract, which causes the contract to lose 10 tokens in such a transaction.

Reference: https://thoreum-finance.medium.com/what-exploit-happened-today-for-gocerberus-and-garuda-also-for-lokum-ybear-piggy-caramelswap-3943ee23a39f

```
553         token.safeTransferFrom(msg.sender, address(this), _amount);
```

- Transferring tokens by `_amount`.

```
563         _deposit(msg.sender, _amount);
```

- This function call executes the following operation.
- In `BUSDChain._deposit`,
  - `_pollDeposits(_addr, _amount);`
- In `BUSDChain._pollDeposits`,
  - `pool_users_refs_deposits_sum[pool_cycle][upline] += _amount;`
- The `_amount` appears to be used for bookkeeping purposes without compensating the potential transfer fees.

Recommendation

We advise the client to regulate the set of tokens supported and add necessary mitigation mechanisms to keep track of accurate balances if there is a need to support deflationary tokens.

Alleviation

[Busdchain Team]:

The project will be worked only based on BUSD tokens and will never be used for deflationary tokens.

BUC-06 | UNCHECKED ERC-20 `transfer()` / `transferFrom()` CALL

Category	Severity	Location	Status
Volatile Code	Minor	BUSDChain.sol (1): 865	Acknowledged

Description

The return value of the transfer()/transferFrom() call is not checked.

```
865         power.transfer(_addr, _amount);
```

Recommendation

Since some ERC-20 tokens return no values and others return a `bool` value, they should be handled with care. We advise using the OpenZeppelin's `SafeERC20.sol` implementation to interact with the `transfer()` and `transferFrom()` functions of external ERC-20 tokens. The OpenZeppelin implementation checks for the existence of a return value and reverts if `false` is returned, making it compatible with all ERC-20 token implementations.

Alleviation

[Busdchain Team]:

Issue acknowledged. I won't make any changes for the current version.

BUC-07 | MISSING THE `whenNotPaused` MODIFIER

Category	Severity	Location	Status
Logical Issue	Minor	BUSDChain.sol (1): 544	Acknowledged

Description

The `withdraw()` function has the `whenNotPaused` modifier which checks that the contract is unpaused. If the owner pauses the contract, the caller can call the `deposit()` function but not the `withdraw()` function. Their tokens will be locked in the contract.

Recommendation

We advise the client to add the `whenNotPaused` modifier to the function.

Alleviation

[Busdchain Team]:

It is one of the rules of the project.

BUC-08 | MISSING ZERO ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	Minor	BUSDChain.sol (1): 520	Acknowledged

Description

The `_upline` address is missing a check that it is not `address(0)`. If the `address(0)` is in the whitelist, the caller can register the `address(0)` as the upline. The caller considers he is already registered, then he calls the function `deposit()`, but is prompted "You need to register first".

Recommendation

We recommend adding a check the passed-in address is not `address(0)` to prevent unexpected errors.

Alleviation

[Busdchain Team]:

The owner is informed. the owner will never add address(0) to the whitelist.

BUC-09 | POTENTIAL LOSS OF POOL BONUS

Category	Severity	Location	Status
Logical Issue	Minor	BUSDChain.sol (1): 579	Acknowledged

Description

The `wL`, `wMY`, and `dfWallets` can not get the pool bonus because the initial state of the `pool_exclude` is `true`. If the owner sets their status to `false`, then they can get the pool bonus, but the function `withdraw()` does not handle the logic about the pool bonus, so they may lose the pool bonus.

Recommendation

We advise the client to check if this can happen.

Alleviation

[Busdchain Team]:

It is one of the rules of the project; the owner is informed.

BUC-10 | 'WL', 'WMY' AND `dfWallets` CAN NOT WITHDRAW DEPOSIT PAYOUT

Category	Severity	Location	Status
Logical Issue	Minor	BUSDChain.sol (1): 566	Acknowledged

Description

In the function `withdraw()`, the 'wL', 'wMY' and `dfWallets` can only get the direct and match bonus. However, there is no restriction in the function `deposit()` that these addresses cannot deposit. If they call the function `deposit()`, they will lose the deposit payout.

Recommendation

We would like to know how to avoid.

Alleviation

[Busdchain Team]:

It is one of our rules; we informed all users of the mentioned wallets.

BUC-11 POTENTIAL WHITELIST CAN NOT WITHDRAW BONUSES AS UPLINE

Category	Severity	Location	Status
Logical Issue	Minor	BUSDChain.sol (1): 566	Acknowledged

Description

In the function `register()`, the whitelist can become upline without the deposit amount, the whitelist will get the direct and pool bonus when the users call the function `deposit()` or get the match bonus when the users call the function `withdraw()`. Due to the deposit amount being 0, then the max payout is 0. So the whitelist calls the function `withdraw()` will revert because it cannot pass the check on line 598.

Recommendation

We advise the client to check this case.

Alleviation

[Busdchain Team]:

It is one of our rules, and all community and whitelist users are informed about it. They need to make deposits to have access to their available dividends.

BUC-12 | HARDCODE ADDRESS

Category	Severity	Location	Status
Logical Issue	Informational	BUSDChain.sol (1): 289~290, 332~335, 337~348	Acknowledged

Description

There are many hardcode addresses in this codebase.

Recommendation

We advise double check the addresses before the contract is deployed onto the blockchain.

Alleviation

No alleviation.

BUC-13 | AN EXTRA LOOP IS EXECUTED

Category	Severity	Location	Status
Logical Issue	Informational	BUSDChain.sol (1): 450~452	Acknowledged

Description

The loop is to move the data after index j forward, but one more loop is executed.

```
450     for(uint256 k = j; k <= pool_size; k++) {
451         pool_top[k] = pool_top[k + 1];
452     }
```

Recommendation

We advise the client to use `k < pool_size` instead of `k <= pool_size`;

Alleviation

No alleviation.

BUC-14 | DISCUSSION : THE USE OF THE POWER TOKEN

Category	Severity	Location	Status
Logical Issue	Informational	BUSDChain.sol (1): 197	Acknowledged

Description

From the contract, the power tokens are used as follows:

1. When the users deposit the `busd` token, they can get the five times the power tokens through the function `deposit()`
2. when the users withdraw, the power tokens are sent to the superior as a reward through the function `withdraw()`
3. If the `burnStatus` is true , the owner of the power tokens can burn them through the function `burnPower()`
4. Owner can transfer the power tokens to any address through the function `powerTransfer()`

There is no logic related to power tokens in the contract, are the power tokens being used as intended and are there other uses for the power tokens?

Recommendation

We would like to confirm with the client if the current implementation aligns with the original project design.

Alleviation

[Busdchain Team]:

The POWER token will be used on our future roadmap plans, and we have bonus and reward features for our other projects via POWER token.

OPTIMIZATIONS | BUSDCHAIN.COM

ID	Title	Category	Severity	Status
BUC-15	Loop Optimization	Gas Optimization	Optimization	Acknowledged

BUC-15 | LOOP OPTIMIZATION

Category	Severity	Location	Status
Gas Optimization	Optimization	BUSDChain.sol (1): 574~577	Acknowledged

Description

On the function `withdraw()`, the loop that checks whether the sender is on the `dfWallets` could be more efficient if it aborted the loop once it has found a match.

Recommendation

We advise the team to add a `break` statement if a match was found.

Alleviation

No alleviation.

APPENDIX | BUSDCHAIN.COM

Finding Categories

Categories	Description
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.

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.

CertiK | Securing the Web3 World

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.