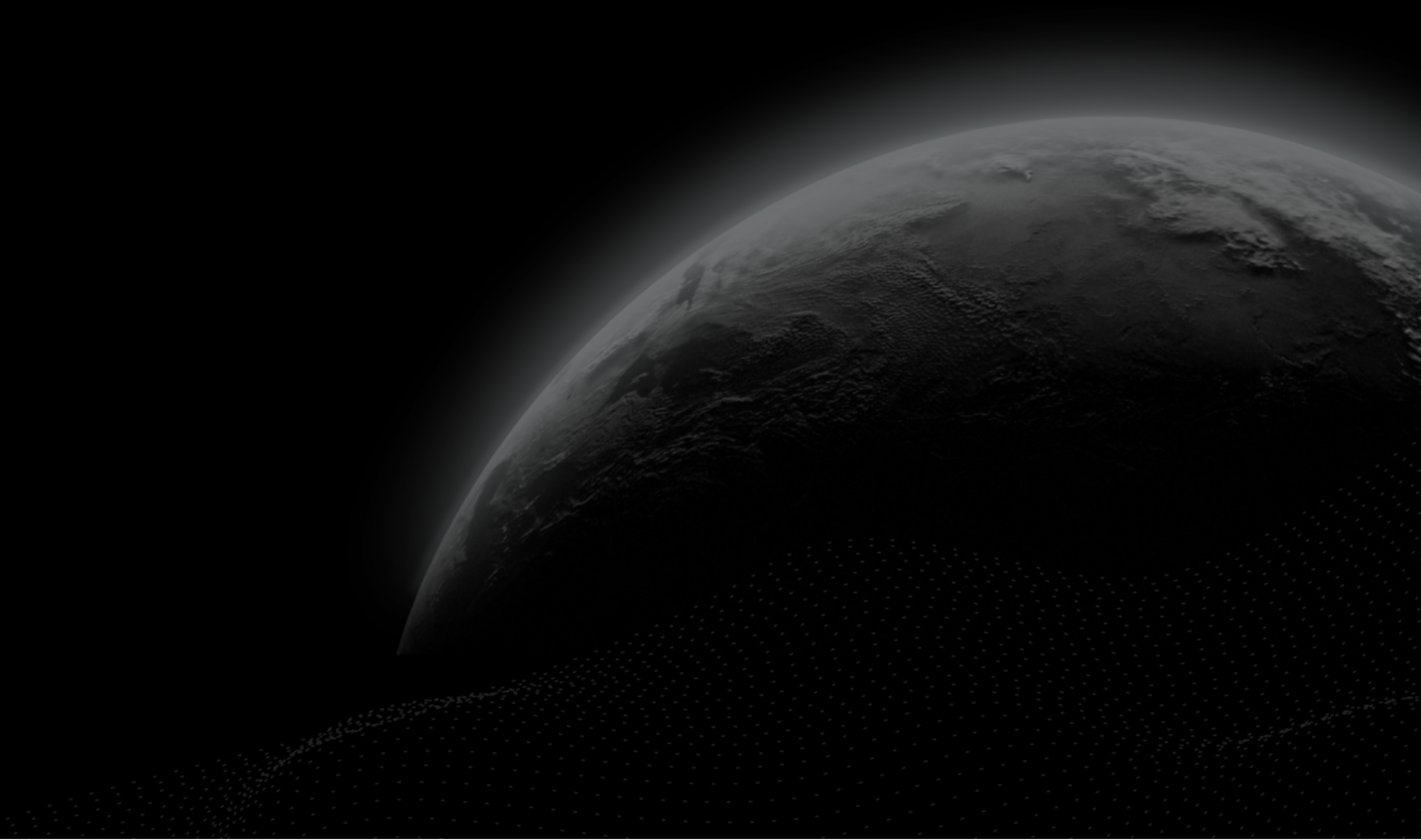
Preliminary Comments

Cyberconnect - Token Bridge

CertiK Assessed on Jun 18th, 2024



CertiK Assessed on Jun 18th, 2024

Cyberconnect - Token Bridge

These preliminary comments were prepared by CertiK, the leader in Web3.0 security.

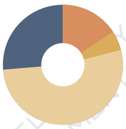
Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Ethereum (ETH)	Manual Review, Static Analysis
LANGUAGE	**TIMELINE**	**KEY COMPONENTS**
Solidity	Delivered on 06/18/2024	N/A

CODEBASE
https://github.com/cyberconnecthq/cyber-token-bridges/
View All in Codebase Page

COMMITS
CyberTokenAdapter.sol, CyberTokenController.sol, LaunchTokenWithdrawer.sol
View All in Codebase Page

Vulnerability Summary



19	4	0	2	12	0	1
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined	Pending

🟥	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.
🟧	3	Major	3 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.
🟨	1	Medium	1 Acknowledged	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
🟨	10	Minor	2 Resolved, 2 Partially Resolved, 6 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.
🟦	5	Informational	2 Resolved, 2 Acknowledged, 1 Pending	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.
⬜	0	Discussion		The impact of the issue is yet to be determined, hence requires further clarifications from the project team.

TABLE OF CONTENTS | CYBERCONNECT - TOKEN BRIDGE

Summary

Findings

Optimizations

Appendix

Disclaimer

CODEBASE | CYBERCONNECT - TOKEN BRIDGE

Repository

https://github.com/cyberconnecthq/cyber-token-bridges/

Commit

CyberTokenAdapter.sol, CyberTokenController.sol, LaunchTokenWithdrawer.sol, RewardTokenWithdrawer.sol:

Base: ed9d401cbc19cca10985263915f7c639e9e66131

CyberStakingPool.sol:

Base: 8b5331c400b18d41577209c53132a63ee03d417f

CyberVault.sol:

Base: 8b5331c400b18d41577209c53132a63ee03d417f

Update: 024293d406023df23bd29537ff99b7f16994e6b4

RewardDistribution.sol:

Base: a376007970c70dde7609e20b6e4e1299d302c9e3

Update: 7db2049287b43062cd40a6c70c0ed1e22dd8730f

AUDIT SCOPE | CYBERCONNECT - TOKEN BRIDGE

7 files audited ● 1 file with Pending findings ● 6 files with Acknowledged findings

ID	Repo	File	SHA256 Checksum
● CVU	cyberconnecthq/cyber-token-bridges	src/CyberVault.sol	bac62c114bc7d3357418ebc574cc76a777779bb6e655157c8528b6b697e3428a
● CTA	cyberconnecthq/cyber-token-bridges	src/CyberTokenAdapter.sol	32c33ab85df67cabaf25e9d108d85a87c6c2b3a72c2651d79d7a942adf4c6388
● CTC	cyberconnecthq/cyber-token-bridges	src/CyberTokenController.sol	22c4a8f577df2e9f3fbdb7f77a32075c7a06016182437e859980a760fa86fd1f
● LTW	cyberconnecthq/cyber-token-bridges	src/LaunchTokenWithdrawer.sol	c649a4665b0a4dce54152b65f0346a3797c71c6a680de3c39a928ce93a7abfbb
● RTW	cyberconnecthq/cyber-token-bridges	src/RewardTokenWithdrawer.sol	ba85e19562376ed048305199baa385cf874e8f6b5d223477a66746224a349133
● CYB	cyberconnecthq/cyber-token-bridges	src/CyberStakingPool.sol	850d0aa2795ee53a26cc6322568d488818f2094fd5794e511cda55579a527d49
● RDB	cyberconnecthq/cyber-token-bridges	src/base/RewardDistribution.sol	522671d8bf98133254bf1747ef06c6e18180ece9c92bf77bed4595b977529603

APPROACH & METHODS | CYBERCONNECT - TOKEN BRIDGE

This report has been prepared for Cyberconnect to discover issues and vulnerabilities in the source code of the Cyberconnect - Token Bridge 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.

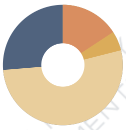
The auditing process pays special attention to the following considerations:

- 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 | CYBERCONNECT - TOKEN BRIDGE



19	0	3	1	10	5	0
Total Findings	Critical	Major	Medium	Minor	Informational	Discussion

This report has been prepared to discover issues and vulnerabilities for Cyberconnect - Token Bridge. Through this audit, we have uncovered 19 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
CYB-01	**Centralization Related Risks**	**Centralization**	**Major**	● **Acknowledged**
RDB-02	**Centralization Risks In RewardDistribution.Sol**	**Centralization**	**Major**	● **Acknowledged**
SRC-03	**Centralized Control Of Contract Upgrade**	**Centralization**	**Major**	● **Acknowledged**
LTW-01	Lack Of Access Control	Access Control	Medium	● Acknowledged
CTC-02	Missing Input Validation	Volatile Code	Minor	● Acknowledged
CVU-02	Unused Pausable Features	Logical Issue	Minor	● Resolved
CYB-03	Incompatibility With Deflationary Tokens (Non-Standard ERC20 Token)	Volatile Code	Minor	● Resolved
CYB-04	Potential Unfair Time Delay For Users When The Time Delay Is Changed	Logical Issue	Minor	● Acknowledged
CYB-05	Missing Limits	Volatile Code	Minor	● Acknowledged
RDB-01	Inconsistency Between Comments And Code	Inconsistency	Minor	● Partially Resolved

ID	Title	Category	Severity	Status
RDB-03	Missing Checks In `RewardDistribution.createDistribution()`	Volatile Code	Minor	● Partially Resolved
RTW-01	Assumptions In `RewardTokenWithdrawer`	Access Control	Minor	● Acknowledged
SRE-01	Out-Of-Scope Dependencies	Volatile Code	Minor	● Acknowledged
SRE-02	Third Party Dependencies	Volatile Code	Minor	● Acknowledged
CVU-03	Potential Inflation Attack Caused By ERC4626 If Implementing Previous OZ Version	Logical Issue	Informational	● Resolved
CVU-04	Vault Implementation Lacks Max Query In Operation	Design Issue	Informational	● Pending
CYB-06	Functions Missing Empty `bytes` Check	Logical Issue	Informational	● Acknowledged
RDB-04	Missing Emit Events	Coding Style	Informational	● Resolved
RTW-03	Potential Missing `payable` Functions In `RewardTokenWithdrawer`	Volatile Code	Informational	● Acknowledged

CYB-01 | CENTRALIZATION RELATED RISKS

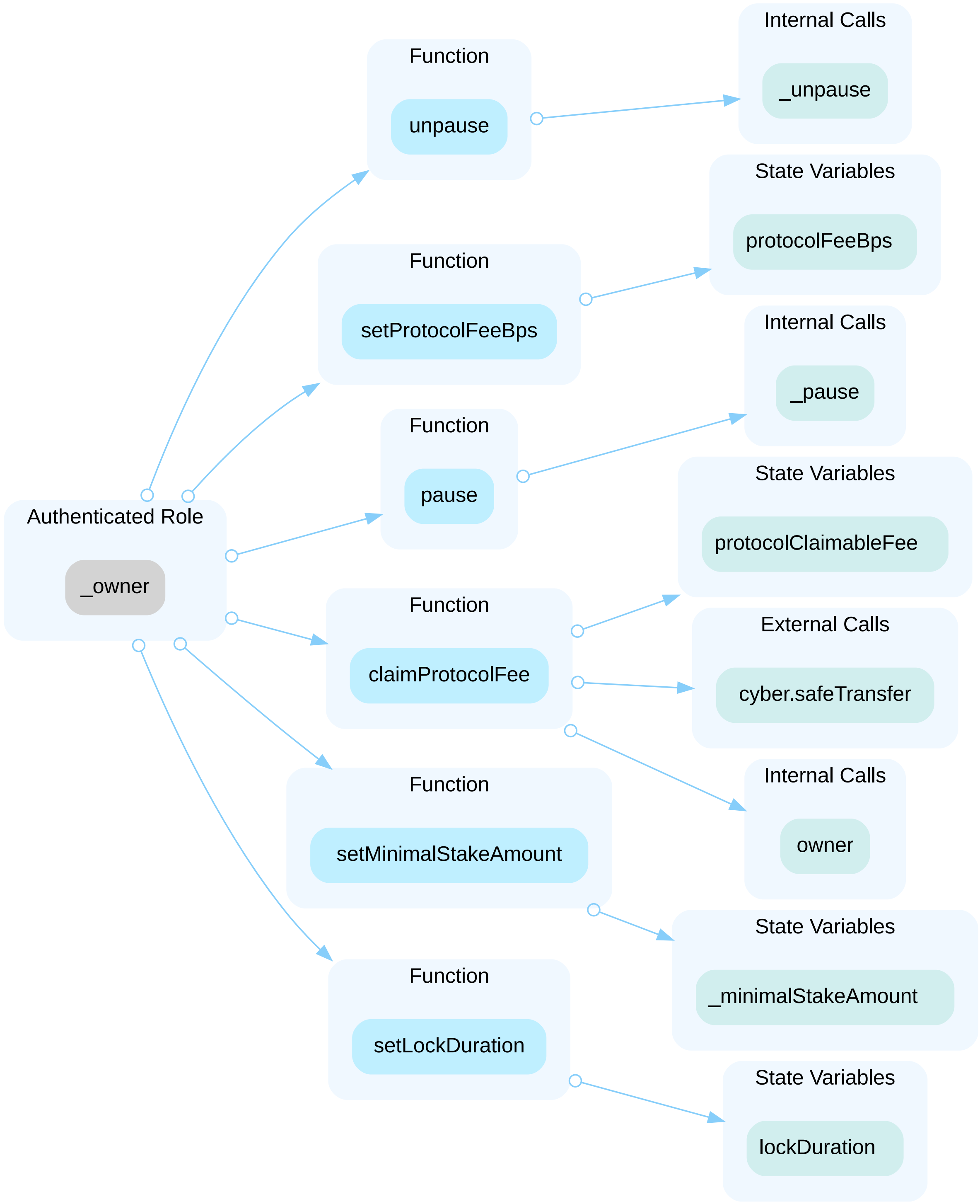
Category	Severity	Location	Status
Centralization	● **Major**	**src/CyberStakingPool.sol (Addendum): 322, 326, 330, 334, 340, 345; src/CyberVault.sol (Addendum): 329, 333, 338; src/CyberTokenController.sol (Base): 113; src/LaunchTokenWithdrawer.sol (Base): 99, 103; src/RewardTokenWithdrawer.sol (Base): 77**	● **Acknowledged**

Description

CyberStakingPool

In the contract `CyberStakingPool` 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, for example:

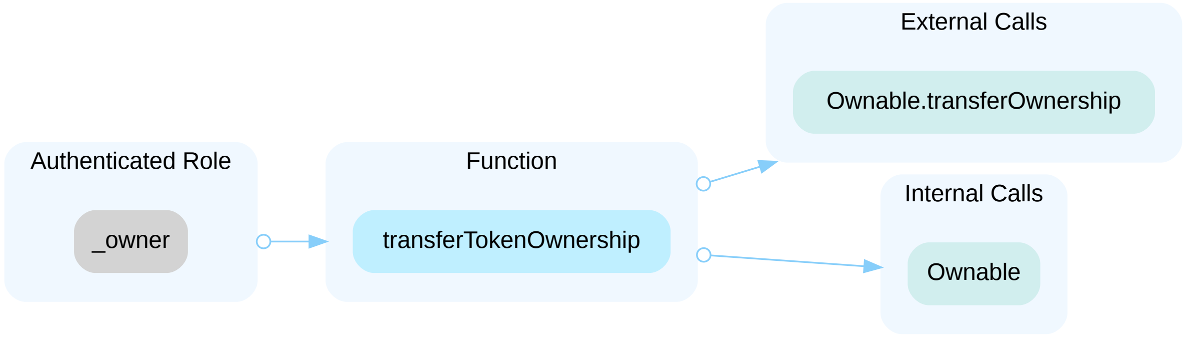
- transfer ownership to an address they control;
- pause or unpause the contract;
- set the protocol fees to any value between 0 and 100%;
- steal the protocol fees;
- set the minimal amount required to stake to an extreme value;
- set the lock duration to an extreme value;



CyberTokenController

In the contract `CyberTokenController` 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:

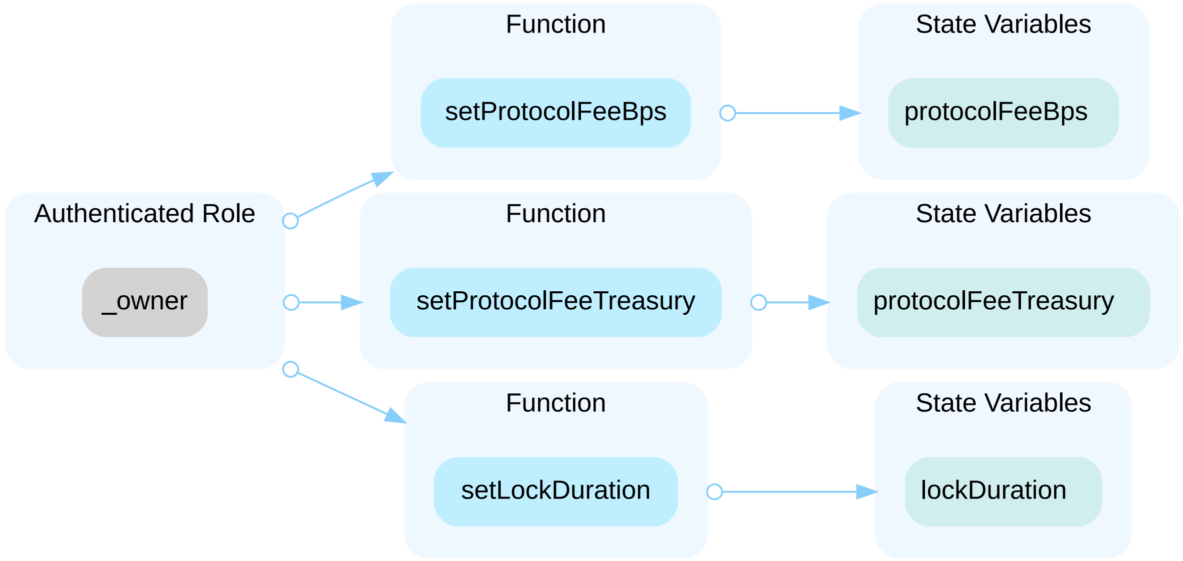
- transfer ownership of `CyberTokenController` to another address they control;
- transfer ownership of `innerToken` to an address they control;



CyberVault

In the contract `CyberVault` 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, for example:

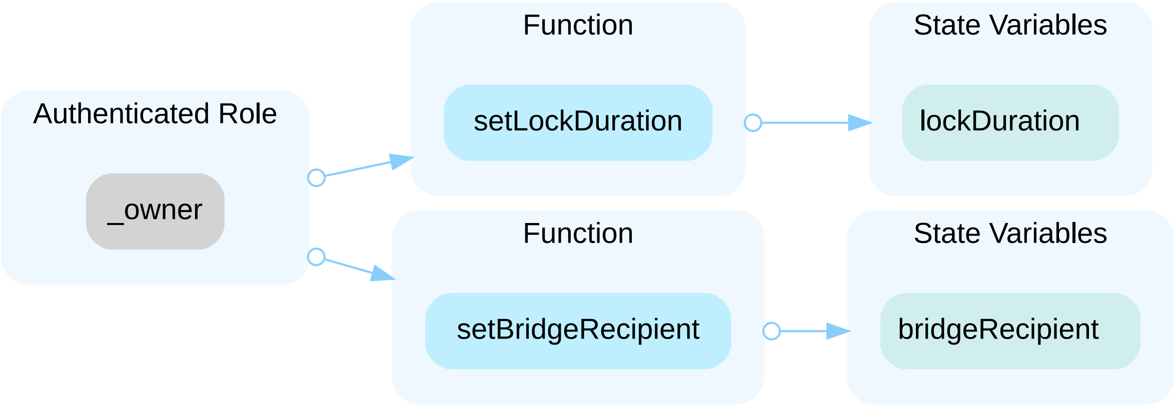
- transfer ownership to an address they control;
- set the protocol fees to any value between 0 and 100%;
- set an address they control as the treasury address;
- set the duration time as an extreme value;



LaunchTokenWithdrawer

In the contract `LaunchTokenWithdrawer` 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, for example:

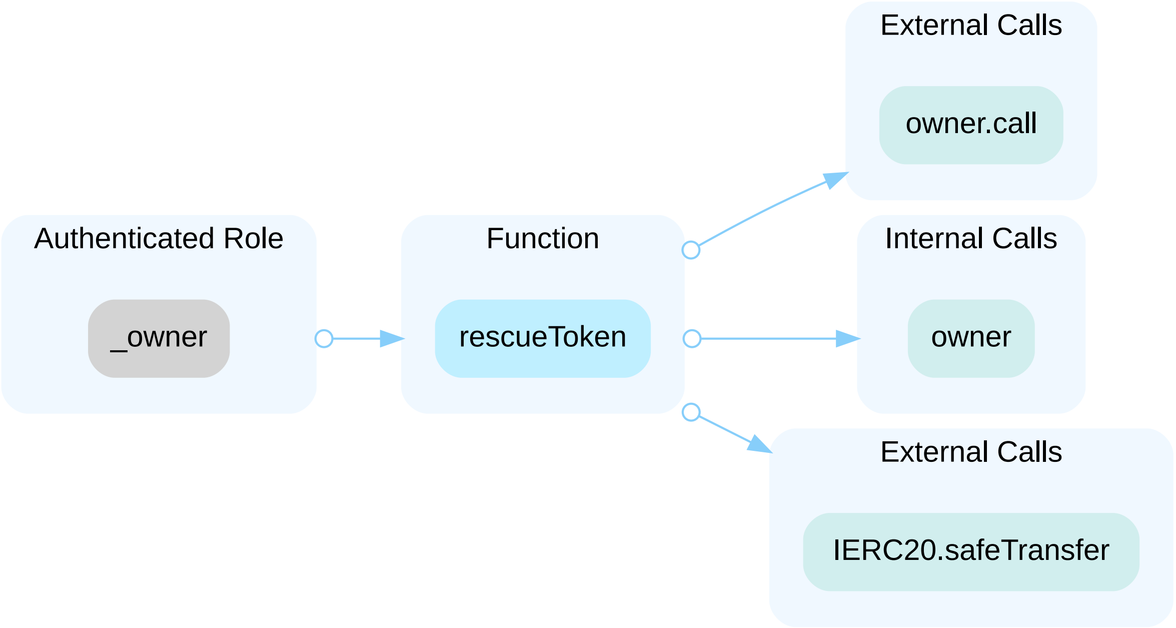
- transfer ownership to an address they control;
- change the `bridgeRecipient` ;
- set the `lockDuration` to an extreme value so the funds are stuck in the contract;



RewardTokenWithdrawer

In the contract `RewardTokenWithdrawer` 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, for example:

- transfer ownership to an address they control;
- steal all the tokens in 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

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

[CertiK, 2024/06/18]: In the new commit 024293d406023df23bd29537ff99b7f16994e6b4, the contract `CyberVault` has now pausable feature, the role `_owner` has now authority over the functions :

- `pause()` ;
- `unpause()` ;

Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and, for example:

- pause or unpause the contract.

RDB-02 | CENTRALIZATION RISKS IN REWARDDISTRIBUTION.SOL

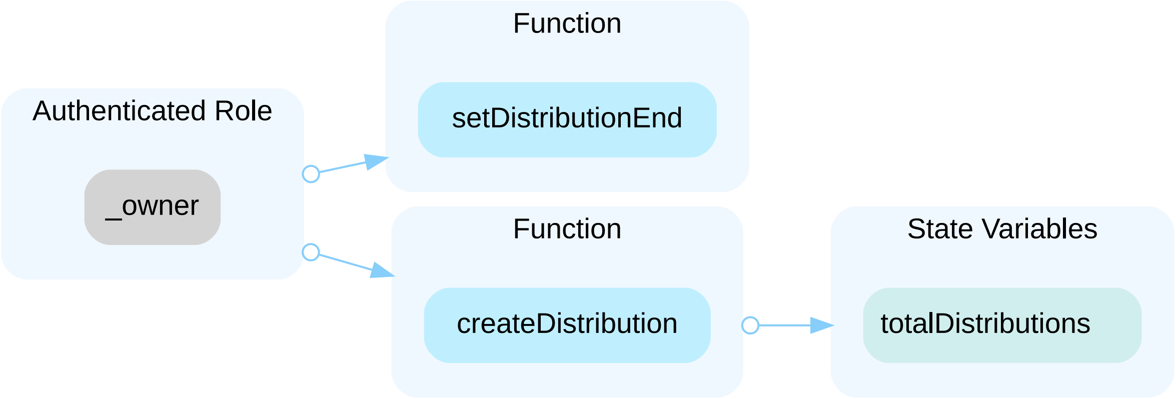
Category	Severity	Location	Status
Centralization	● Major	src/base/RewardDistribution.sol (Addendum2): 41, 64	● Acknowledged

Description

RewardDistribution.sol

In the contract `RewardDistribution` 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, for example:

- transfer ownership to an address they control;
- set the end of a specific distribution to an arbitrary future time, which is irreversible;
- creating new, unwanted distributions, disrupting the expected model;



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

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

SRC-03 | CENTRALIZED CONTROL OF CONTRACT UPGRADE

Category	Severity	Location	Status
Centralization	● **Major**	**src/CyberStakingPool.sol (Addendum): 21; src/CyberVault.sol (Addendum): 27**	● **Acknowledged**

Description

In contracts `CyberStakingPool` and `CyberVault` , the role `owner` has the authority to upgrade the implementation contract.

Any compromise to the `owner` account may allow a hacker to take advantage of this authority and change the implementation contract which is pointed by proxy and therefore execute potential malicious functionality in the implementation contract.

Recommendation

We recommend that the team make efforts to restrict access to the admin of the proxy contract. A strategy of combining a time-lock and a multi-signature (⅔, ⅗) wallet can be used to prevent a single point of failure due to a private key compromise. In addition, the team should be transparent and notify the community in advance whenever they plan to migrate to a new implementation contract.

Here are some feasible short-term and long-term suggestions that would mitigate the potential risk to a different level and suggestions that would permanently fully resolve the risk.

Short Term:

A combination of a time-lock and a multi signature (⅔, ⅗) wallet mitigate the risk by delaying the sensitive operation and avoiding a single point of key management failure.

- A time-lock with reasonable latency, such as 48 hours, for awareness of privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to a private key compromised;
  AND
- A medium/blog link for sharing the time-lock contract and multi-signers addresses information with the community.

For remediation and mitigated status, please provide the following information:

- Provide the deployed time-lock address.
- Provide the **gnosis** address with **ALL** the multi-signer addresses for the verification process.

- Provide a link to the **medium/blog** with all of the above information included.

Long Term:

A combination of a time-lock on the contract upgrade operation and a DAO for controlling the upgrade operation mitigate the contract upgrade risk by applying transparency and decentralization.

- A time-lock with reasonable latency, such as 48 hours, for community awareness of privileged operations;
  AND
- Introduction of a DAO, governance, or voting module to increase decentralization, transparency, and user involvement;
  AND
- A medium/blog link for sharing the time-lock contract, multi-signers addresses, and DAO information with the community.

For remediation and mitigated status, please provide the following information:

- Provide the deployed time-lock address.
- Provide the **gnosis** address with **ALL** the multi-signer addresses for the verification process.
- Provide a link to the **medium/blog** with all of the above information included.

Permanent:

Renouncing ownership of the `admin` account or removing the upgrade functionality can *fully* resolve the risk.

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

Note: we recommend the project team consider 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

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

LTW-01 | LACK OF ACCESS CONTROL

Category	Severity	Location	Status
Access Control	● Medium	src/LaunchTokenWithdrawer.sol (Base): 75	● Acknowledged

Description

The function `LaunchTokenWithdrawer.withdraw()` can be called by anyone as it has no access restriction.

Recommendation

We recommend adding a restriction on potential callers.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

CTC-02 | MISSING INPUT VALIDATION

Category	Severity	Location	Status
Volatile Code	● Minor	src/CyberTokenController.sol (Base): 24~25, 29~30	● Acknowledged

Description

The constructor of `CyberTokenController` is missing a check that `_delegate` is not `address(0)` nor `_token` .

Recommendation

We recommend adding a check the passed-in address is not `address(0)` nor `_token` to prevent unexpected errors.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

CVU-02 | UNUSED PAUSABLE FEATURES

Category	Severity	Location	Status
Logical Issue	● Minor	src/CyberVault.sol (Addendum): 84~85	● Resolved

Description

In the contract `CyberVault` , the contract `PausableUpgradeable` is imported and initialized, however, the pausing mechanism is never used in this contract.

Recommendation

We recommend using the pausable features or removing the contract.

Alleviation

[CertiK, 2024/06/18]: The client made changes resolving the finding in commit 024293d406023df23bd29537ff99b7f16994e6b4.

CYB-03 INCOMPATIBILITY WITH DEFLATIONARY TOKENS (NON-STANDARD ERC20 TOKEN)

Category	Severity	Location	Status
Volatile Code	● Minor	src/CyberStakingPool.sol (Addendum): 189	● Resolved

Description

The project design may not be compatible with non-standard ERC20 tokens, such as deflationary tokens or rebase tokens.

The functions use `transferFrom()` / `transfer()` to move funds from the sender to the recipient but fail to verify if the received token amount matches the transferred amount. This could pose an issue with fee-on-transfer tokens, where the post-transfer balance might be less than anticipated, leading to balance inconsistencies. There might be subsequent checks for a second transfer, but an attacker might exploit leftover funds (such as those accidentally sent by another user) to gain unjustified credit.

Scenario

When transferring deflationary ERC20 tokens, the input amount may not equal 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 arrive 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.

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 non-standard ERC20 tokens.

Alleviation

[Cyberconnect team, 2024/06/14]: "CYBER is not a deflationary token".

CYB-04 | POTENTIAL UNFAIR TIME DELAY FOR USERS WHEN THE TIME DELAY IS CHANGED

Category	Severity	Location	Status
Logical Issue	● Minor	src/CyberStakingPool.sol (Addendum): 208	● Acknowledged

Description

The protocol utilizes a state variable `lockDuration` to set the mandatory waiting period for users to unstake or withdraw funds. Altering `lockDuration` between staking transactions could introduce discrepancies, leading to an unfair waiting period for users, which may extend the time for some users more than others.

Scenario

Suppose the `lockDuration` is updated from `6 weeks` to `1 day`. In that case, the previous users must wait for 6 weeks to perform the `unstake` operation, while the newcomers only need to wait for 1 day, even if they are just a few blocks apart during `staking`. This might cause confusion among the users.

Recommendation

We recommend capturing the `block.timestamp` upon each staking action and utilizing `block.timestamp + lockDuration` for comparison, ensuring `lockDuration` remains consistent for all users throughout their waiting period.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

CYB-05 | MISSING LIMITS

Category	Severity	Location	Status
Volatile Code	● Minor	src/CyberStakingPool.sol (Addendum): 330~331, 334~335; src/CyberVault.sol (Addendum): 329~330, 329~330, 338~339; src/LaunchTokenWithdrawer.sol (Base): 99~100, 103~104	● Acknowledged

Description

CyberVault

- `setLockDuration()` allows setting any arbitrary value as the `lockDuration` .

CyberStakingPool

- `setLockDuration()` allows setting any arbitrary value as the `lockDuration` ;
- `setMinimalStakeAmount()` allows setting any arbitrary value as the `_minimalStakeAmount` .

LaunchTokenWithdrawer

- `setLockDuration()` allows setting any arbitrary value as the `lockDuration` .

Recommendation

We recommend adding lower and upper limits to prevent these values from being set too high or too low.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

RDB-01 | INCONSISTENCY BETWEEN COMMENTS AND CODE

Category	Severity	Location	Status
Inconsistency	● Minor	src/base/RewardDistribution.sol (Addendum2): 101~102	● Partially Resolved

Description

The comments of `RewardDistribution._getDistributionIndex()` state that one of the parameters is meant to be the `totalSupply` :

```
101      /// @param totalSupply The total token supply.
```

However, this parameter is meant to be the return value from `CyberStakingPool.circulatingSupply()` which is

```
292          return totalSupply() - protocolLockedAmount;
```

Recommendation

We recommend clarifying the expected value of the `_getDistributionIndex()` parameter and :

- modifying the comment if it does not describe the function accurately, or
- modifying the function's logic so the input parameter reflects the total token supply.

Additionally, providing documentation and the expected specifications of the related functions would help determine the parameters' accuracy.

Alleviation

[CertiK, 2024/06/18]: The client made changes partially resolving the finding in commit 7db2049287b43062cd40a6c70c0ed1e22dd8730f.
The following issues still exist:

- no documentation or expected specifications were provided.

RDB-03 MISSING CHECKS IN `RewardDistribution.createDistribution()`

Category	Severity	Location	Status
Volatile Code	● Minor	src/base/RewardDistribution.sol (Addendum2): 58~59	● Partially Resolved

Description

In `RewardDistribution.createDistribution()`, `distribution.emissionPerSecond` can be freely set by the `owner`. This could be an issue since setting it to zero would prevent any incrementation of the `distribution.userIndices[]` for any user. Also if a value is mistakenly set too high, this could cause the index to be incremented too fast.

The end time of a distribution has no restriction on the maximum value allowed. This also applies to `RewardDistribution.setDistributionEnd()`.

This poses a potential risk of mistakenly creating a distribution with an excessively distant end time, which could lead to unexpected behavior.

Recommendation

We recommend adding reasonable upper and lower bounds to prevent setting incorrect values as important parameters of a distribution.

Alleviation

[CertiK, 2024/06/18]: The client made changes partially resolving the finding in commit 7db2049287b43062cd40a6c70c0ed1e22dd8730f.
The following issues still exist:

- missing upper limit on `distribution.emissionPerSecond`;
- missing upper limit on `distribution.endTime`.

RTW-01 | ASSUMPTIONS IN `RewardTokenWithdrawer`

Category	Severity	Location	Status
Access Control	● Minor	src/RewardTokenWithdrawer.sol (Base): 56~57	● Acknowledged

Description

In the contract `RewardTokenWithdrawer` , the functions `withdraw()` and `stake()` trigger the transfer of tokens or shares to an address given as input by the caller. The only safety check relies on `MerkleDistribution._consumeProof()` which is outside the scope of this audit.

The assumptions on the verification mechanisms are:

- **Collision Resistance**: it is not possible to use two different inputs producing the same hash.
- **Unforgeability**: it is impossible to produce a valid Merkle proof with invalid data.
- **Non-reusability**: each Merkle proof can only be used once.
- **Parameter Integrity**: The proof is tied to specific transaction data and parameters, ensuring that any alteration to these elements will invalidate the proof.
- **Confidentiality**: it is not possible to deduce another user's Merkle proof.

Recommendation

We recommend verifying all the assumptions described above are well implemented to prevent any exploit due to an implementation or design mistake.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

SRE-01 | OUT-OF-SCOPE DEPENDENCIES

Category	Severity	Location	Status
Volatile Code	● Minor	src/CyberStakingPool.sol (Base): 13~14, 26~27, 354~355; src/CyberTokenAdapter.sol (Base): 5~6, 8, 13~14; src/CyberTokenController.sol (Base): 8~9, 12~13, 26~27, 78~79; src/LaunchTokenWithdrawer.sol (Base): 8~9, 14~15, 53~54; src/RewardTokenWithdrawer.sol (Base): 9~10, 15~16, 41~42	● Acknowledged

Description

The protocol is serving as the underlying entity to interact with out-of-scope dependencies. The out-of-scope dependencies that the contracts interact with are:

- `OFTAdapter.sol` ;
- `RewardDistribution.sol` ;
- `OFTCore.sol` ;
- `MerkleDistribution.sol` ;

Recommendation

We recommend all out-of-scope dependencies are carefully vetted to ensure they function as intended. Last, we recommend all assumptions about the behavior of the project are thoroughly reviewed and, if the assumptions do not match the intention of the protocol, documenting the intended behavior for review.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

SRE-02 | THIRD PARTY DEPENDENCIES

Category	Severity	Location	Status
Volatile Code	● Minor	src/CyberTokenController.sol (Base): 10~11, 84~85, 104~105; src/CyberVault.sol (Base): 15~16, 58~59, 89~90	● Acknowledged

Description

The protocol is serving as the underlying entity to interact with third-party protocols. The third parties that the contracts interact with are:

- `innerToken` through the interface `IMintableBurnable` ;
- `cyberStakingPool` through the interface `ICyberStakingPool` ;

The scope of the audit treats third-party entities as black boxes and assumes their functional correctness. However, in the real world, third parties can be compromised and this may lead to lost or stolen assets. Moreover, updates to the state of a project contract that are dependent on the read of the state of external third-party contracts may make the project vulnerable to read-only reentrancy. In addition, upgrades of third parties can possibly create severe impacts, such as increasing fees of third parties, migrating to new LP pools, etc.

Recommendation

We recommend constantly monitoring the third parties involved to mitigate any side effects that may occur when unexpected changes are introduced, as well as vetting any third-party contracts used to ensure no external calls can be made before updates to its state. Additionally, we recommend all out-of-scope dependencies are carefully vetted to ensure they function as intended. Last, we recommend all assumptions about the behavior of the project are thoroughly reviewed and, if the assumptions do not match the intention of the protocol, documenting the intended behavior for review.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

CVU-03 | POTENTIAL INFLATION ATTACK CAUSED BY ERC4626 IF IMPLEMENTING PREVIOUS OZ VERSION

Category	Severity	Location	Status
Logical Issue	● Informational	src/CyberVault.sol (Addendum): 28~29	● Resolved

Description

The `CyberVault` contract inherits `ERC4626` contracts from Openzeppelin. If the version is below `v4.9`, when the vault is empty or nearly empty, deposits are at high risk of being stolen through frontrunning with a "donation" to the vault that inflates the price of a share. This is well-known as a donation or inflation attack and is essentially a slippage problem.

Recommendation

We recommend using a version above `v4.9`. If this is not possible, to mitigate this issue, it is recommended that vault deployers make an initial deposit, atomic with deployment of the contract, of a non-trivial amount of the asset, such that price manipulation becomes infeasible.

Alleviation

[Cyberconnect team, 2024/06/14]: "We are using ^5.0.1 now."

CVU-04 VAULT IMPLEMENTATION LACKS MAX QUERY IN OPERATION

Category	Severity	Location	Status
Design Issue	● Informational	src/CyberVault.sol (Addendum): 166, 184	● Pending

Description

The vault contract, which is based on the `ERC4626` tokenized vault standard, is required to comply with the rules set forth by the `EIP4626` standard for vaults. The standard includes specific guidelines for handling user interactions predictably and safely, particularly concerning the maximum transaction sizes that can be processed by the contract without failure.

As per the `EIP4626` standard, the vault contract must implement the following functions: `maxDeposit` , `maxMint` , `maxWithdraw` , and `maxRedeem` . Each of these functions is responsible for returning the maximum amount that can be processed by their corresponding `deposit` , `mint` , `withdraw` , and `redeem` functions without causing a revert (an error that undoes all changes made during the transaction).

The vault contract should be designed to consult these `max-` functions prior to executing any `deposit` , `mint` , `withdraw` , or `redeem` operations. By querying the `max-` function, the contract can determine if the intended operation is within the current acceptable range. This comparison is crucial to prevent transaction failures. If the intended transaction amount exceeds the value returned by the `max-` function, the contract should not proceed with the operation, as it would inevitably lead to a revert.

Recommendation

To mitigate this issue, it is recommended to:

Integrate checks within the `deposit` , `mint` , `withdraw` , and `redeem` functions to call their respective `max-` functions before executing the transaction. If the intended transaction amount is greater than the value returned by the `max-` function, the operation should not be executed, and an appropriate error message should be returned to the user.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version.

I think we have max query."

[CeriK, 2024/06/18]: The contract includes max query functions, which are directly inherited from the OpenZeppelin library and utilize the default parameters provided by OpenZeppelin. To ensure these parameters align with your expectations and specifications, could you please confirm that the default parameters meet your requirements?

CYB-06 | FUNCTIONS MISSING EMPTY `bytes` CHECK

Category	Severity	Location	Status
Logical Issue	● Informational	src/CyberStakingPool.sol (Addendum): 221~235; src/RewardTokenWithdrawer.sol (Base): 50~59	● Acknowledged

Description

Passing empty bytes to a function can cause unexpected behavior, such as certain operations failing, producing incorrect results, or wasting gas.

Recommendation

It is recommended to check that all `bytes` parameters are not empty.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version".

RDB-04 | MISSING EMIT EVENTS

Category	Severity	Location	Status
Coding Style	● Informational	src/base/RewardDistribution.sol (Addendum2): 41, 64	● Resolved

Description

There should always be events emitted in the sensitive functions that are controlled by centralization roles.

Recommendation

It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

Alleviation

[CertiK, 2024/06/18]: The client made changes resolving the finding in commit 7db2049287b43062cd40a6c70c0ed1e22dd8730f.

RTW-03 | POTENTIAL MISSING `payable` FUNCTIONS IN `RewardTokenWithdrawer`

Category	Severity	Location	Status
Volatile Code	● Informational	src/RewardTokenWithdrawer.sol (Base): 79	● Acknowledged

Description

The contract `RewardTokenWithdrawer` , does not have any `payable` function, but has a function `rescueToken()` allowing the transfer of native tokens to another address.

Recommendation

We recommend adding `receive` functions if the contract is meant to handle native tokens.

Alleviation

[Cyberconnect team, 2024/06/14]: "Issue acknowledged. I won't make any changes for the current version.

The contract is not designed to receive native tokens. But the rescueToken is a common way for ownable contract to rescue tokens".

OPTIMIZATIONS | CYBERCONNECT - TOKEN BRIDGE

ID	Title	Category	Severity	Status
CVU-01	Missing Validation On Zero Address	Logical Issue	Optimization	● Resolved

CVU-01 | MISSING VALIDATION ON ZERO ADDRESS

Category	Severity	Location	Status
Logical Issue	● Optimization	src/CyberVault.sol (Addendum): 139, 153, 167	● Resolved

Description

Parameters of type address in your functions should be checked to ensure that they are not assigned the null address (address(0x0)). Failure to validate these parameters can lead to transaction reverts, wasted gas, the need for transaction resubmission, and may even require redeployment of contracts within the protocol in certain situations. Implement checks for address(0x0) to avoid these potential issues.

Recommendation

We recommend validate the address are not assigned the null address (address(0x0)).

Alleviation

[CertiK, 2024/06/18]: The client made changes resolving the finding in commit e4f8823d046289e018ee7fc6b446f95aa1b8d32c.

APPENDIX | CYBERCONNECT - TOKEN BRIDGE

Finding Categories

Categories	Description
Coding Style	Coding Style findings may not affect code behavior, but indicate areas where coding practices can be improved to make the code more understandable and maintainable.
Access Control	Access Control findings are about security vulnerabilities that make protected assets unsafe.
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.
Design Issue	Design Issue findings indicate general issues at the design level beyond program logic that are not covered by other finding categories.

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.