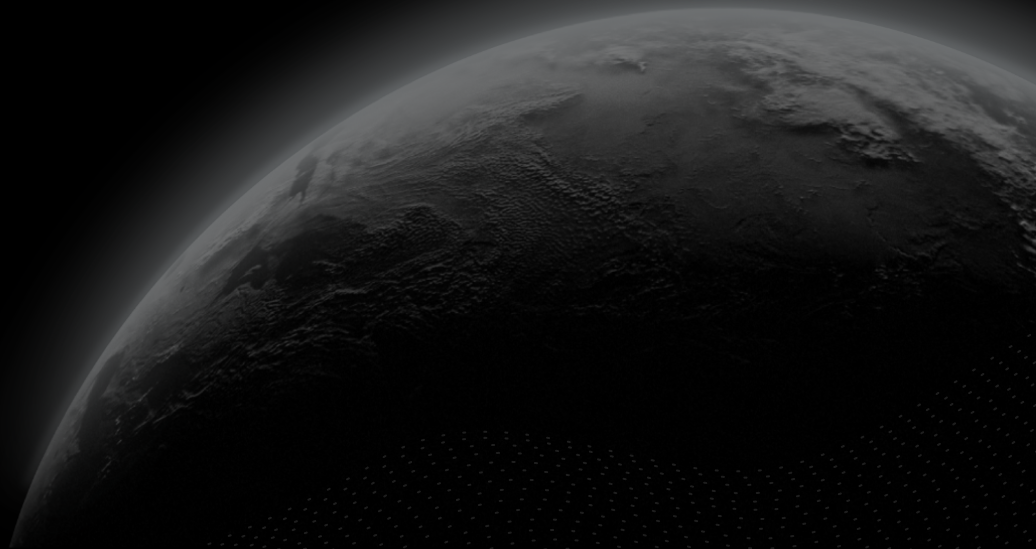




Security Assessment

ArchiSwap

CertiK Assessed on Jul 7th, 2023





Certik Assessed on Jul 7th, 2023

ArchiSwap

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

Executive Summary

TYPES

DeFi

ECOSYSTEM

zkSync

METHODS

Manual Review, Static Analysis

LANGUAGE

Solidity

TIMELINE

Delivered on 07/07/2023

KEY COMPONENTS

N/A

CODEBASE

- <https://explorer.zksync.io/address/0xdC182d95d1876BB03B48146C85a748ae748d9137#contract>
- <https://goerli.explorer.zksync.io/address/0x9277dB69143dCa9>

[View All in Codebase Page](#)

Vulnerability Summary



14

Total Findings

3

Resolved

0

Mitigated

0

Partially Resolved

11

Acknowledged

0

Declined

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 Resolved



Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.

6 Minor

1 Resolved, 5 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.

4 Informational

4 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 | ARCHISWAP

I **Summary**

[Executive Summary](#)

[Vulnerability Summary](#)

[Codebase](#)

[Audit Scope](#)

[Approach & Methods](#)

I **Review Notes**

[Overview](#)

[External Dependencies](#)

I **Decentralization Efforts**

[Privileged Functions](#)

I **Findings**

[CKP-02 : Centralization Risks](#)

[CKP-03 : Centralized Control of Upgradeable Contracts](#)

[ASC-06 : Inconsistency between K validation and K update](#)

[ASC-07 : Possible Inaccurate Return Value](#)

[ASC-03 : Potential Cross-Chain Replay Attack](#)

[ASC-04 : Divide Before Multiply](#)

[ASC-12 : Unchecked ERC-20 `transfer\(\)`/`transferFrom\(\)` Call](#)

[ASC-13 : Missing Input Validation](#)

[ASC-14 : Possible Conflicts When Setting Configuration](#)

[CKP-01 : Missing Zero Address Validation](#)

[ASC-15 : Incorrect Error Message](#)

[ASC-16 : Use of `ecrecover\(\)`](#)

[ASC-17 : Test Code Should be Removed](#)

[PCK-02 : The functions can be initialized multiple times](#)

I **Appendix**

I **Disclaimer**



CODEBASE | ARCHISWAP

Repository

- <https://explorer.zksync.io/address/0xdC182d95d1876BB03B48146C85a748ae748d9137#contract>
- <https://goerli.explorer.zksync.io/address/0x9277dB69143dCa96f987531ed02B9c8CDcdcd505#contract>
- <https://explorer.zksync.io/address/0x3271625e618525Db504f4C4f9EBf757bB76d6Baa#contract>

AUDIT SCOPE | ARCHISWAP

2 files audited ● 2 files with Acknowledged findings

ID	File	SHA256 Checksum
● ASC	 ArchiSwap.sol	11d66b6d7f4d18fcef8bf72b379885bf6e57ca b8bc450f143e76abb9dbad44e
● PCK	 Proxy0516.sol	a75bad678b5c07a3f86a852a57fb821c42306 24ef07285e19c16138c3e9dfaaf

APPROACH & METHODS | ARCHISWAP

This report has been prepared for ArchiSwap to discover issues and vulnerabilities in the source code of the ArchiSwap 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.

REVIEW NOTES | ARCHISWAP

Overview

Archiswap is a decentralized exchange DEX built on zkSync. It also incorporates farming functionality, thereby allowing users to earn returns on their assets. By swapping tokens using Archiswap, users can earn tokens as a reward, creating a compelling incentive structure.

Note: The contracts `ArchiswapFarmingRouter`, `DeployFactory`, and `DeployRouter` in the original codebase are considered out of scope and have been removed in the most recent codebase (July 7, 2023).

External Dependencies

Archiswap inherits or uses a few of the depending injection contracts or addresses to fulfill the need of its business logic. 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.

- Contract `UpgradeabilityProxy`: `_logic`.
- Contract `ProductProxy`, `InitializableProductProxy`: `_factory`, `_tokens`, `token`.
- Contract `__BaseAdminUpgradeabilityProxy__`: `newImplementation`.
- Contract `InitializableUpgradeabilityProxy`: `_logic`.
- Contract `ArchiswapPair`: `factory`, `token0`, `token1`, `to`.
- Contract `ArchiswapFactory`: `pair`.
- Contract `ArchiswapRouter02`: `factory`, `pair`, `WETH`, `input`, `path`.

DECENTRALIZATION EFFORTS | ARCHISWAP

Privileged Functions

- In the contract `ArchiSwapFactory` the role `_setter` has authority over the following functions:
- `setProductImplementation()` to set new implementation contract.
- `setFeeTo()` to set the `feeTo` address.
- `setTaxTo()` to set the `taxTo` address.
- `setFeeToSetter()` to set the `feeToSetter` address.
- `setFeeRate()` to set the fee rate and tax rate for the input `pair`.

In the contract `Configurable` the role `governor` has authority over the following functions:

- `setConfig()` to set config information.
- `setConfigI()` to set config with the index.
- `setConfigA()` to set config with the address.

In the contract `Governable` the role `governor` has authority over the following functions:

- `renounceGovernorship()` to allow the current governor to relinquish control of the contract.
- `transferGovernorship()` to allow the current governor to transfer control of the contract to a new governor.

In the contract `InitializableProductProxy`, the roles `factory_`, `governor`, and `admin` have authority over the following functions:

- `__InitializableProductProxy_init()` to initialize the contract.

In the contract `__InitializableAdminUpgradeabilityProductProxy__` the role `admin_` has authority over following the function:

- `__InitializableAdminUpgradeabilityProductProxy_init__()` to initialize the contract.

In the contracts `BaseAdminUpgradeabilityProxy` and `__BaseAdminUpgradeabilityProxy__`, the role `admin` has authority over the following functions:

- `admin()` to return the address of the proxy admin.
- `implementation()` to return the address of the implementation.
- `changeAdmin()` to change the admin of the proxy.
- `upgradeTo()` to upgrade the backing implementation of the proxy.
- `upgradeToAndCall()` to upgrade the backing implementation of the proxy and call a function.
- `__changeAdmin__()` to change the admin of the proxy.

- `__upgradeTo__()` to upgrade the backing implementation of the proxy.
- `__upgradeToAndCall__` to upgrade the backing implementation of the proxy and call a function.

Any compromise to the privileged account may allow the hacker to take advantage of this authority and manipulate the state variables. The potential impact could disrupt the platform's stability, create unfair trading condition, and expose users to higher risks or encourage risky trading behavior, leading to significant financial losses for users.

To improve the trustworthiness of the project, dynamic runtime updates in the project should be notified to the community. Any plan to invoke the aforementioned functions should also be considered to move to the execution queue of `TimeLock` contract.

FINDINGS | ARCHISWAP



14

Total Findings

0

Critical

2

Major

2

Medium

6

Minor

4

Informational

This report has been prepared to discover issues and vulnerabilities for ArchiSwap. Through this audit, we have uncovered 14 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
CKP-02	Centralization Risks	Centralization	Major	● Acknowledged
CKP-03	Centralized Control Of Upgradeable Contracts	Centralization	Major	● Acknowledged
ASC-06	Inconsistency Between K Validation And K Update	Logical Issue	Medium	● Resolved
ASC-07	Possible Inaccurate Return Value	Logical Issue	Medium	● Resolved
ASC-03	Potential Cross-Chain Replay Attack	Logical Issue	Minor	● Acknowledged
ASC-04	Divide Before Multiply	Incorrect Calculation	Minor	● Acknowledged
ASC-12	Unchecked ERC-20 <code>transfer()</code> / <code>transferFrom()</code> Call	Volatile Code	Minor	● Acknowledged
ASC-13	Missing Input Validation	Volatile Code	Minor	● Resolved
ASC-14	Possible Conflicts When Setting Configuration	Logical Issue	Minor	● Acknowledged
CKP-01	Missing Zero Address Validation	Volatile Code	Minor	● Acknowledged
ASC-15	Incorrect Error Message	Coding Style	Informational	● Acknowledged

ID	Title	Category	Severity	Status
ASC-16	Use Of <code>ecrecover()</code>	Design Issue	Informational	● Acknowledged
ASC-17	Test Code Should Be Removed	Coding Issue	Informational	● Acknowledged
PCK-02	The Functions Can Be Initialized Multiple Times	Coding Issue	Informational	● Acknowledged

CKP-02 | CENTRALIZATION RISKS

Category	Severity	Location	Status
Centralization	● Major	ArchiSwap.sol: 102, 111, 152, 155, 158, 664, 668, 672, 677, 681; Proxy0516.sol: 170~171, 177~178, 186~187, 197~198, 210~211, 366~367, 380~381, 398~399	● Acknowledged

Description

In the contract `ArchiSwapFactory` the role `_setter` has authority over the following functions:

- `setProductImplementation()` to set new implementation contract.
- `setFeeTo()` to set the `feeTo` address.
- `setTaxTo()` to set the `taxTo` address.
- `setFeeToSetter()` to set the `feeToSetter` address.
- `setFeeRate()` to set the fee rate and tax rate for the input `pair`.

In the contract `Configurable` the role `governor` has authority over the following functions:

- `setConfig()` to set config information.
- `setConfigI()` to set config with the index.
- `setConfigA()` to set config with the address.

In the contract `Governable` the role `governor` has authority over the following functions:

- `renounceGovernorship()` to allow the current governor to relinquish control of the contract.
- `transferGovernorship()` to allow the current governor to transfer control of the contract to a new governor.

In the contract `InitializableProductProxy`, the roles `factory_`, `governor`, and `admin` have authority over the following functions:

- `__InitializableProductProxy_init()` to initialize the contract.

In the contract `__InitializableAdminUpgradeabilityProductProxy__` the role `admin_` has authority over following the function:

- `__InitializableAdminUpgradeabilityProductProxy_init__()` to initialize the contract.

In the contracts `BaseAdminUpgradeabilityProxy` and `__BaseAdminUpgradeabilityProxy__`, the role `admin` has authority over the following functions:

- `admin()` to return the address of the proxy admin.
- `implementation()` to return the address of the implementation.
- `changeAdmin()` to change the admin of the proxy.
- `upgradeTo()` to upgrade the backing implementation of the proxy.
- `upgradeToAndCall()` to upgrade the backing implementation of the proxy and call a function.
- `__changeAdmin__()` to change the admin of the proxy.
- `__upgradeTo__()` to upgrade the backing implementation of the proxy.
- `__upgradeToAndCall__` to upgrade the backing implementation of the proxy and call a function.

Any compromise to the privileged account may allow the hacker to take advantage of this authority and manipulate the state variables. The potential impact could disrupt the platform's stability, create unfair trading condition, and expose users to higher risks or encourage risky trading behavior, leading to significant financial losses for users.

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 (2/3, 3/5) 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.

I Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue and will not make any changes for the current version.

[CertiK, June 28, 2023]: The ArchiSwap team project has yet to address the centralization related risks. They plan to handle this issue at a later stage of development.

CKP-03 | CENTRALIZED CONTROL OF UPGRADEABLE CONTRACTS

Category	Severity	Location	Status
Centralization	● Major	ArchiSwap.sol: 67~68, 273~274, 367~368, 431~432, 1099~1100, 1564~1565; Proxy0516.sol: 197~198, 380~381, 441~442, 493~494, 619~620	● Acknowledged

Description

Below contracts are upgradeable:

- `Governable` .
- `UniswapV2ERC20` .
- `ArchiSwapPair` .
- `ArchiSwapRouter02` .
- `ArchiSwapFarmingRouter` .
- `InitializableAdminUpgradeabilityProxy` .
- `__AdminUpgradeabilityProxy__` .
- `__InitializableAdminUpgradeabilityProductProxy__` .

The owner can upgrade the contract and set the new implementation contract without the community's commitment. If an attacker compromises the account, it can change the state of the contract and drain tokens from the 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 (2/3, 3/5) 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 (2/3, 3/5) wallet mitigates 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-signer 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 mitigates 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-signer 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.

I Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue and will not make any changes for the current version.

[CertiK, June 28, 2023]: The ArchiSwap team project has yet to address the centralization related risks. They plan to handle this issue at a later stage of development.

ASC-06 | INCONSISTENCY BETWEEN K VALIDATION AND K UPDATE

Category	Severity	Location	Status
Logical Issue	● Medium	ArchiSwap.sol: 559~569	● Resolved

Description

After performing the K validation, a small amount of `token0` or `token1` is transferred to `_taxTo` if either of them is designated as `_taxToken`. Let's say `token0` is the `_taxToken`. In this case, the actual `balance0` will be less than the value that was used during the K validation process.

The K validation is used to ensure the pool has possesses an adequate number of tokens to maintain liquidity. When taxes are set excessively high, a significant discrepancy arises between the balance values before and after K validation. This discrepancy leads to inaccurate pricing for `price0CumulativeLast` and `price1CumulativeLast`. Consequently, adding liquidity becomes unfeasible because the `mint()` function cannot be executed correctly due to the incorrect price.

Recommendation

Recommend the client review the code logic and make necessary changes to the K validation to avoid any expected errors.

Alleviation

[ArchiSwap, June 27, 2023]: The K value is verified by `balance0Adjusted` and `balance1Adjusted`, and the fee has been subtracted (and `_taxRate` is included in the fee, which is a certain percentage of the fee,) so this problem should not exist.

[Certik, June 29, 2023]: It's crucial to ensure that `_taxRate` is set to a value less than `1e18`. If it exceeds this limit, the `tax` computed will be higher than the fees calculated in subsequent code snippet. This could potentially disrupt the logical flow and lead to unexpected results.

```
553         uint balance0Adjusted = balance0.mul(1e18).sub(amount0In.mul(_feeRate))
      / 1e14;
554         //uint balance1Adjusted = balance1.mul(1000).sub(amount1In.mul(3));
555         uint balance1Adjusted = balance1.mul(1e18).sub(amount1In.mul(_feeRate))
      / 1e14;
```

[ArchiSwap, July 7, 2023]: The issue is resolved at address [0x3271625e618525Db504f4C4f9EBf757bB76d6Baa](https://etherscan.io/address/0x3271625e618525Db504f4C4f9EBf757bB76d6Baa) where limits are set on the fee and tax rates.

ASC-07 | POSSIBLE INACCURATE RETURN VALUE

Category	Severity	Location	Status
Logical Issue	● Medium	ArchiSwap.sol: 799, 810, 1424	● Resolved

Description

The functions `getAmountsOut()` and `getAmountsIn()` calls `getAmountOut()`, respectively `getAmountIn()`, when determining swap amounts. The functions `getAmountOut()` and `getAmountIn()` call `getAmountOutPair()` and `getAmountInPair()` using `address(0)` as the `pair` parameter.

```
773     function getAmountOut(uint amountIn, uint reserveIn, uint reserveOut)
public view returns (uint amountOut) {
774         return getAmountOutPair(address(0), amountIn, reserveIn, reserveOut);
775     }
```

This means that the default fees are used for calculating swap amounts. As these functions are used by the router, there may be inaccurate swaps if a token in the swap path has a custom fee different from the default fee.

If a custom fee is lower than the default fee, users may acquire less tokens than expected as the swap assumes a higher fee is present. In contrast, if a custom fee is higher than the default fee, swaps may revert as more tokens than expected need to be used for a swap.

The issue also occurs in `_swapSupportingFeeOnTransferTokens()` in the router, where `getAmountOut()` is directly used.

Recommendation

It is recommended to not use the default fee when computing swap amounts, but rather use the actual fee.

Alleviation

[ArchiSwap, June 27, 2023]: The team heeded the advice and resolved the issue in new url

[0x4649A58dE162b6ED5F650AFC51D8FC9FE0cC0Eac](https://github.com/ArchiSwap/0x4649A58dE162b6ED5F650AFC51D8FC9FE0cC0Eac) by directly calling

`ArchiSwapFactory(factory).getAmountOutPair()` and `ArchiSwapFactory(factory).getAmountInPair()` instead.

ASC-03 | POTENTIAL CROSS-CHAIN REPLAY ATTACK

Category	Severity	Location	Status
Logical Issue	Minor	ArchiSwap.sol: 355~359, 361	Acknowledged

Description

Signed messages are not properly verified with the current chain ID, thus allowing attackers to perform replay attacks across chains. Hardcoded or cached chain ID values are also vulnerable since a hard fork may occur and change the chain ID in the future.

```
355         abi.encodePacked(  
356             '\x19\x01',  
357             DOMAIN_SEPARATOR,  
358             keccak256(abi.encode(PERMIT_TYPEHASH, owner, spender, value,  
nonces[owner]++, deadline))  
359         )
```

- Reading a state variable `DOMAIN_SEPARATOR`, which seems to use a cached chain ID value.
- Encoding the cached chain ID value `DOMAIN_SEPARATOR`.

```
361         address recoveredAddress = ecrecover(digest, v, r, s);
```

- Calling `ecrecover` with a hash that may include an outdated chain ID.

Recommendation

We recommend verifying signed messages against the current chain ID by using `block.chainid` or `chainid()` within the same transaction.

Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue. `DOMAIN_SEPARATOR` contains `chainId`. Unless there is a chain with the same `chainId`, replay attacks are possible. If the chain forks in the future, the team will upgrade the contract.

ASC-04 | DIVIDE BEFORE MULTIPLY

Category	Severity	Location	Status
Incorrect Calculation	● Minor	ArchiSwap.sol: 467, 553, 555, 557, 561, 566, 767, 768, 1681, 1687, 1691, 1692, 1738, 1740	● Acknowledged

Description

Performing integer division before multiplication truncates the low bits, losing the precision of calculation.

```
467          uint numerator = (totalSupply.mul(_taxRate) / 1e14).mul(
rootK.sub(rootKLast));
```

```
553          uint balance0Adjusted = balance0.mul(1e18).sub(amount0In.mul(_feeRate))
/ 1e14;
```

```
555          uint balance1Adjusted = balance1.mul(1e18).sub(amount1In.mul(_feeRate))
/ 1e14;
```

```
557          require(balance0Adjusted.mul(balance1Adjusted) >= uint(_reserve0).mul(
_reserve1).mul(1e8), 'UniswapV2: K');
```

```
561          tax = (tax.mul(_feeRate) / 1e18).mul(_taxRate) / 1e18;
```

```
566          tax = (tax.mul(_feeRate) / 1e18).mul(_taxRate) / 1e18;
```

```
767          uint amountInWithFee = amountIn.mul(uint(1e18).sub(_feeRate)) / 1e14;
```

```
768          uint numerator = amountInWithFee.mul(reserveOut);
```

```
1681          tax = amount.mul(rate).div(1e18);
```

```
1691          reward = rwdBuf.mul(tax).div(taxsBuf);
```

```
1687         taxesBuf = taxesBuf.div(now.sub(begin[taxToken]));
```

```
1692         taxesBuf = taxesBuf.mul(period[taxToken]).div(period[taxToken].add(now)
    .sub(lastUpdateTime[taxToken]));
```

```
1738         amt = amt.mul(now.sub(lastUpdateTime[taxToken])).div(
    rewardsDuration[taxToken]);
```

```
1740         amt = amt.mul(now.sub(lastUpdateTime[taxToken])).div(periodFinish
    [taxToken].sub(lastUpdateTime[taxToken]));
```

Recommendation

We recommend applying multiplication before division to avoid loss of precision.

Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue.

The team all writes code according to the rule of multiplying first and then dividing. At the same time, the team should not multiply more than 2 times in a row to prevent overflow. Generally, it is multiplied once and divided once.

ASC-12 | UNCHECKED ERC-20 `transfer()` / `transferFrom()` CALL

Category	Severity	Location	Status
Volatile Code	Minor	ArchiSwap.sol: 1204	Acknowledged

Description

The return values of the `transfer()` and `transferFrom()` calls in the smart contract are not checked. Some ERC-20 tokens' transfer functions return no values, while others return a bool value, they should be handled with care. If a function returns `false` instead of reverting upon failure, an unchecked failed transfer could be mistakenly considered successful in the contract.

```
1204      IUniswapV2Pair(pair).transferFrom(msg.sender, pair, liquidity);  
// send liquidity to pair
```

Recommendation

It is advised to use 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

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue. The pair here is determined by our contract, `transfer()` and `transferFrom()` must return bool.

ASC-13 | MISSING INPUT VALIDATION

Category	Severity	Location	Status
Volatile Code	Minor	ArchiSwap.sol: 672~673, 681~682	Resolved

Description

It is crucial to validate the linked parameters to ensure they fall within the acceptable value ranges.

```
674      taxPriority[_taxToken] = _taxPriority;
```

```
682      feeRate[pair] = _feeRate;  
683      taxRate[pair] = _taxRate;
```

According to the following code snippets:

```
553      uint balance0Adjusted = balance0.mul(1e18).sub(amount0In.mul(_feeRate))  
    / 1e14;
```

```
467      uint numerator = (totalSupply.mul(_taxRate) / 1e14).mul(rootK.sub(  
    rootKLast));
```

The parameters `_feeRate` and `_taxRate` ought to be valued less than `1e18`.

Recommendation

Recommend the client review the code logic to define the valid range for the input values and add necessary checks to ensure the input is valid.

Alleviation

[ArchiSwap, July 7, 2023]: The issue is resolved at address [0x3271625e618525Db504f4C4f9EBf757bB76d6Baa](https://etherscan.io/address/0x3271625e618525Db504f4C4f9EBf757bB76d6Baa) where limits are set on the fee and tax rates.

ASC-14 | POSSIBLE CONFLICTS WHEN SETTING CONFIGURATION

Category	Severity	Location	Status
Logical Issue	● Minor	ArchiSwap.sol: 153, 156, 159	● Acknowledged

Description

The contract `Configurable` has three functions to change configurations: `setConfig()`, `setConfigI()`, and `setConfigA()`.

```
152     function setConfig(bytes32 key, uint value) external governance {
153         _setConfig(key, value);
154     }
155     function setConfigI(bytes32 key, uint index, uint value) external
governance {
156         _setConfig(bytes32(uint(key) ^ index), value);
157     }
158     function setConfigA(bytes32 key, address addr, uint value) public
governance {
159         _setConfig(bytes32(uint(key) ^ uint(addr)), value);
160     }
```

However, there are no restrictions as to what `key` can be used in each function, causing potential conflicts. For example, a key `keyA` used in `setConfigI` may overlap with a key `keyB` used in `setConfig` depending on the `index`.

Recommendation

It is recommended to design the keys used in each configuration function in a way such that conflicts will not occur.

Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue. The first few values are set by the administrator, which are implemented after careful inspection.

CKP-01 | MISSING ZERO ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	● Minor	ArchiSwap.sol: 80, 339~340, 344~345, 435, 436, 618, 665, 669, 677, 1115, 1116, 1609, 1610; Proxy0516.sol: 290~291, 453~454, 507~508, 627~628	● Acknowledged

Description

Addresses are not validated before assignment or external calls, potentially allowing the use of zero addresses and leading to unexpected behavior or vulnerabilities. For example, transferring tokens to a zero address can result in a permanent loss of those tokens.

```
80      governor = governor_;
```

- `governor_` is not zero-checked before being used.

```
339      _transfer(msg.sender, to, value);
```

- `to` is not zero-checked before being used.

```
344      function transferFrom(address from, address to, uint value) external  
returns (bool) {
```

- `to` is not zero-checked before being used.

```
435      token0 = _token0;
```

- `_token0` is not zero-checked before being used.

```
436      token1 = _token1;
```

- `_token1` is not zero-checked before being used.

```
618      productImplementation = _productImplementation;
```

- `_productImplementation` is not zero-checked before being used.

```
665      productImplementation = _productImplementation;
```

- `_productImplementation` is not zero-checked before being used.

```
669      feeTo = _feeTo;
```

- `_feeTo` is not zero-checked before being used.

```
678      feeToSetter = _feeToSetter;
```

- `_feeToSetter` is not zero-checked before being used.

```
1115     factory = _factory;
```

- `_factory` is not zero-checked before being used.

```
1116     WETH = _WETH;
```

- `_WETH` is not zero-checked before being used.

```
1609     rewardsDistribution = _rewardsDistribution;
```

- `_rewardsDistribution` is not zero-checked before being used.

```
1610     ecoAddr = _ecoAddr;
```

- `_ecoAddr` is not zero-checked before being used.

```
266 (bool success,) = _logic.delegatecall(_data);
```

- `_logic` is not zero-checked before being used.

```
400 (bool success,) = newImplementation.delegatecall(data);
```

- `newImplementation` is not zero-checked before being used.

```
481 (bool success,) = _logic.delegatecall(_data);
```

- `_logic` is not zero-checked before being used.

The contracts `AdminUpgradeabilityProxy`, `__AdminUpgradeabilityProxy__`, `InitializableAdminUpgradeabilityProxy`, and `__InitializableAdminUpgradeabilityProductProxy__` do not perform a non-zero address validation on the `admin`. If the `admin` is set to the zero address, it cannot be changed to a valid, non-zero address because the `changeAdmin()` function uses the `ifAdmin` modifier. This modifier requires that a valid `admin` is present to modify the existing `admin`. Once the `admin` has been assigned via the contract constructor, it becomes impossible to change the `admin` value.

Recommendation

It is recommended to add a zero-check for the passed-in address value to prevent unexpected errors.

Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue. These addresses are set by the administrator, and the settings are implemented after careful inspection.

ASC-15 | INCORRECT ERROR MESSAGE

Category	Severity	Location	Status
Coding Style	● Informational	ArchiSwap.sol: 411~412, 441~442, 495~496, 516~517, 530~531, 532~533, 539~540, 548~549, 557~558, 628~629, 630~631, 631~632, 1142~1143, 1147~1148, 1208~1209, 1323~1324, 1337~1338, 1351~1352, 1353~1354, 1365~1366, 1367~1368, 1382~1383, 1384~1385, 1400~1401, 1402~1403, 1443~1446, 1460~1461, 1466~1469, 1483~1484, 1489~1490	● Acknowledged

Description

The error messages referenced in the provided code snippets indicate `UniswapV2`, however, the contract currently in operation is named `Archi`.

```
411         require(success && (data.length == 0 || abi.decode(data, (bool))),
'UniswapV2: TRANSFER_FAILED');
```

```
441         require(balance0 <= uint112(-1) && balance1 <= uint112(-1),
'UniswapV2: OVERFLOW');
```

```
495         require(liquidity > 0, 'UniswapV2: INSUFFICIENT_LIQUIDITY_MINTED');
```

```
516         require(amount0 > 0 && amount1 > 0,
'UniswapV2: INSUFFICIENT_LIQUIDITY_BURNED');
```

```
530         require(amount0Out > 0 || amount1Out > 0,
'UniswapV2: INSUFFICIENT_OUTPUT_AMOUNT');
531         ...
532         require(amount0Out < _reserve0 && amount1Out < _reserve1,
'UniswapV2: INSUFFICIENT_LIQUIDITY');
533
```

```
539         require(to != _token0 && to != _token1, 'UniswapV2: INVALID_TO');
```

```
548         require(amount0In > 0 || amount1In > 0,
'UniswapV2: INSUFFICIENT_INPUT_AMOUNT');
```

```
557         require(balance0Adjusted.mul(balance1Adjusted) >= uint(_reserve0).mul(
_reserve1).mul(1e8), 'UniswapV2: K');
```

```
628         require(tokenA != tokenB, 'UniswapV2: IDENTICAL_ADDRESSES');
629         ...
630         require(token0 != address(0), 'UniswapV2: ZERO_ADDRESS');
631         require(getPair[token0][token1] == address(0), 'UniswapV2: PAIR_EXISTS'
); // single check is sufficient
```

```
1106         require(deadline >= block.timestamp, 'UniswapV2Router: EXPIRED');
```

```
1142         require(amountBOptimal >= amountBMin,
'UniswapV2Router: INSUFFICIENT_B_AMOUNT');
1143         ...
1144         require(amountAOptimal >= amountAMin,
'UniswapV2Router: INSUFFICIENT_A_AMOUNT');
```

```
1208         require(amountA >= amountAMin,
'UniswapV2Router: INSUFFICIENT_A_AMOUNT');
1209         require(amountB >= amountBMin,
'UniswapV2Router: INSUFFICIENT_B_AMOUNT');
```

```
1323         require(amounts[amounts.length - 1] >= amountOutMin,
'UniswapV2Router: INSUFFICIENT_OUTPUT_AMOUNT');
1324
```

```
1337         require(amounts[0] <= amountInMax,
'UniswapV2Router: EXCESSIVE_INPUT_AMOUNT');//@audit error message
```

```
1351         require(path[0] == WETH, 'UniswapV2Router: INVALID_PATH');
1352         ...
1353         require(amounts[amounts.length - 1] >= amountOutMin,
'UniswapV2Router: INSUFFICIENT_OUTPUT_AMOUNT');
```

```
1365         require(path[path.length - 1] == WETH,
'UniswapV2Router: INVALID_PATH');
1366         ...
1367         require(amounts[0] <= amountInMax,
'UniswapV2Router: EXCESSIVE_INPUT_AMOUNT');
```

```
1382         require(path[path.length - 1] == WETH,  
'UniswapV2Router: INVALID_PATH');  
1383         ...  
1384         require(amounts[amounts.length - 1] >= amountOutMin,  
'UniswapV2Router: INSUFFICIENT_OUTPUT_AMOUNT');
```

```
1400         require(path[0] == WETH, 'UniswapV2Router: INVALID_PATH');  
1401         ...  
1402         require(amounts[0] <= msg.value,  
'UniswapV2Router: EXCESSIVE_INPUT_AMOUNT');
```

```
1443         require(  
1444             IERC20(path[path.length - 1]).balanceOf(to).sub(balanceBefore) >=  
amountOutMin,  
1445             'UniswapV2Router: INSUFFICIENT_OUTPUT_AMOUNT'  
1446         );
```

```
1460         require(path[0] == WETH, 'UniswapV2Router: INVALID_PATH');  
1461         ...  
1462         require(  
1463             IERC20(path[path.length - 1]).balanceOf(to).sub(balanceBefore) >=  
amountOutMin,  
1464             'UniswapV2Router: INSUFFICIENT_OUTPUT_AMOUNT'  
1465         );
```

```
1483         require(path[path.length - 1] == WETH,  
'UniswapV2Router: INVALID_PATH');
```

```
1489         require(amountOut >= amountOutMin,  
'UniswapV2Router: INSUFFICIENT_OUTPUT_AMOUNT');
```

Recommendation

Recommend the client make necessary changes to the error messages.

Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue. They will fix the issue in the future, which will not be included in this audit engagement.

ASC-16 | USE OF `ecrecover()`

Category	Severity	Location	Status
Design Issue	● Informational	ArchiSwap.sol: 361	● Acknowledged

Description

The `permit()` function currently only allows signatures compatible with `ecrecover()`. However, zkSync supports account abstraction, which may include other signature schemes. Users who use these other schemes may be unable to use `permit()`.

Reference: <https://era.zksync.io/docs/reference/concepts/aa.html#aa-signature-checker>

Recommendation

It is recommended to accept other signature schemes.

Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue. zkSync currently does not have other signature schemes, so the team will not consider other signature schemes for the time being, and the team will upgrade the contract when there are other signature schemes.

ASC-17 | TEST CODE SHOULD BE REMOVED

Category	Severity	Location	Status
Coding Issue	● Informational	ArchiSwap.sol: 1541~1558	● Acknowledged

Description

The functions `faucet()` and `_faucetInit()`, along with the variables `testTokenZK`, `testTokenSYNC`, and `faucetTime` should be removed before deployment to the mainnet.

Recommendation

It is recommended to remove test functionality.

Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue. This is for the user to experience the product.

PCK-02 | THE FUNCTIONS CAN BE INITIALIZED MULTIPLE TIMES

Category	Severity	Location	Status
Coding Issue	● Informational	Proxy0516.sol: 604~605, 620~621	● Acknowledged

Description

Without any restriction, the linked functions `__InitializableAdminUpgradeabilityProductProxy_init__` and `__InitializableProductProxy_init` can be called multiple times, which might lead to unexpected behavior or security vulnerabilities. It would be good practice to prevent double initialization if these functions are setting initial states that shouldn't be altered later.

Recommendation

We would like to discuss this finding with the team to make sure this design is intended and correct.

Alleviation

[ArchiSwap, June 27, 2023]: The team has acknowledged this issue. The team allows administrators to call `__InitializableAdminUpgradeabilityProductProxy_init__` and `__InitializableProductProxy_init` as many times as they please.

APPENDIX | ARCHISWAP

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.
Coding Issue	Coding Issue findings are about general code quality including, but not limited to, coding mistakes, compile errors, and performance issues.
Incorrect Calculation	Incorrect Calculation findings are about issues in numeric computation such as rounding errors, overflows, out-of-bounds and any computation that is not intended.
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.

