CERTIK

Security Assessment

Dypius - Audit

CertiK Assessed on Oct 31st, 2023

CertiK Assessed on Oct 31st, 2023

Dypius - Audit

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

Executive Summary

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

LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 10/31/2023	N/A

CODEBASE

https://github.com/dypfinance/Dypius-token-bridge-bsc/tree/2617f0b85e3eefc01381602c33e073db6b50b437/Contracts
https://github.com/dypfinance/Dypius-token-bridge-
View All in Codebase Page

COMMITS

2617f0b85e3eefc01381602c33e073db6b50b437
7d6a7489f65ad5593d9facacb6df2dc80fef2a6f
e87628d9f7984c0486c3858a1857c9f17355708a
View All in Codebase Page

Vulnerability Summary

7	3	1	2	1	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	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	1 Mitigated, 1 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.
0	Medium		Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
2	Minor	1 Resolved, 1 Partially Resolved	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
3	Informational	2 Resolved, 1 Partially Resolved	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 | DYPIUS - AUDIT

CODEBASE | DYPIUS - AUDIT

Repository

https://github.com/dypfinance/Dypius-token-bridge-bsc/tree/2617f0b85e3eefc01381602c33e073db6b50b437/Contracts
https://github.com/dypfinance/Dypius-token-bridge-bsc/tree/7d6a7489f65ad5593d9facacb6df2dc80fef2a6f/Contracts

Commit

2617f0b85e3eefc01381602c33e073db6b50b437
7d6a7489f65ad5593d9facacb6df2dc80fef2a6f
e87628d9f7984c0486c3858a1857c9f17355708a

AUDIT SCOPE | DYPIUS - AUDIT

3 files audited ● 1 file with Partially Resolved findings ● 2 files with Resolved findings

ID	Repo	File	SHA256 Checksum
● CDI	dypfinance/Dypius-token-bridge-bsc	dypius_eth.sol	abb4242b5e347b132e12af025a7f38191a40e446834c579ea13ffc8b119de058
● CDU	dypfinance/Dypius-token-bridge-bsc	bridge_dyp_mint.sol	88c570520dd63c0f95f0a4bca87ae9fa5536309280f0c1ccce82295217cce7c3
● CDH	dypfinance/Dypius-token-bridge-bsc	dypius.sol	885e7deae0343ad8b5e7c23d9646574f10157b2d07f76531ca3b9f431fd5825c

APPROACH & METHODS | DYPIUS - AUDIT

This report has been prepared for Dypius to discover issues and vulnerabilities in the source code of the Dypius - Audit 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 | DYPIUS - AUDIT

7	0	2	0	2	3
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Dypius - Audit. Through this audit, we have uncovered 7 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
CDU-03	**Initial Token Distribution**	**Centralization**	**Major**	**Mitigated**
GLOBAL-01	**Centralization Related Risks**	**Centralization**	**Major**	**Acknowledged**
CDI-01	Solidity Version Not Recommended	Language Version	Minor	Partially Resolved
CDU-01	Potential Cross-Chain Replay Attack	Logical Issue	Minor	Resolved
CDU-02	Incompatibility With Deflationary Tokens	Logical Issue	Informational	Resolved
CDU-04	Delegation Not Moved Along With Token-Transfer	Logical Issue	Informational	Partially Resolved
CDU-05	Too Many Digits	Coding Style	Informational	Resolved

CDU-03 | INITIAL TOKEN DISTRIBUTION

Category	Severity	Location	Status
Centralization	**Major**	**dypius_eth.sol (7d6a748): 802**	**Mitigated**

Description

All of the `dypius` tokens are sent to the contract deployer when deploying the contract. This could be a centralization risk as the anonymous deployer can distribute tokens without obtaining the consensus of the community. Any compromise to the deployer account that holds undistributed tokens may allow the attacker to steal and sell tokens on the market, resulting in severe damage to the project.

Recommendation

It's recommended the team be transparent regarding the initial token distribution process. The token distribution plan should be published in a public location that the community can access. The team shall make enough efforts to restrict the access of the private key. A multi-signature (⅔, ⅗) wallet can be used to prevent a single point of failure due to the private key compromise. Additionally, the team can lock up a portion of tokens, release them with a vesting schedule for long-term success, and deanonymize project teams with a third-party KYC provider to create greater accountability.

Alleviation

The team published the token distribution plan, which can be found here:

- https://twitter.com/dypius/status/1715369180348780756
- https://dypius.medium.com/should-dypius-do-a-token-migration-to-upgrade-the-smart-contract-for-the-bridge-and-other-related-a97315841969

The DYP team also provided the following information to further clarify the token distribution plan:

1. The team has developed a smart contract that allows users to swap their old DYP V1 tokens for new ones. The contract can be found here. The old DYP V1 tokens will be automatically burned by this smart contract. Based on a snapshot, the team determined that 34,309,070 DYP V2 tokens need to be claimed by EOA users, excluding the exchanges.

2. The BSC Claim contract can be accessed here. This was the old bridge contract, but it now lacks the 'transferAnyERC20' function that would give the Owner access to the funds, ensuring that the funds remain locked. To claim the tokens, users must use another Bridge contract from the BSC network and deposit their OLD DYP V1 tokens. After this deposit, a 1:1 ratio will be applied for the swap. From a snapshot, the team found that 17,382,470 DYP V2 tokens need to be claimed by EOA users, excluding exchanges.

3. The AVAX Claim contract is available here. Like the BSC Claim contract, it is the old bridge contract without the 'transferAnyERC20' function, so the funds are locked. Users need to utilize another Bridge contract from the AVAX network and deposit their OLD DYP V1 tokens. Once the deposit is made, a 1:1 ratio swap will be executed. According to a snapshot, 17,382,470 DYP V2 tokens are awaiting claim by EOA users, not counting the exchanges.

4. The remaining DYP V2 tokens, amounting to 168,517,311 DYP, will be distributed to CEXs that trade DYP. The team has locked these tokens in a Token Lock Contract available here. The team transferred the Token Lock Ownership to the Timelock contract found here, which is from OpenZeppelin v4.6.0. On the Timelock contract, the team set the Proposer as the Deployer contract of DYP V2 and the Executor as a Multi-sig wallet they created, available here. This multi-sig wallet has three owners, and two signatures are required for a proposal to be executed. Additionally, the timelock contract includes a 3-day delay.

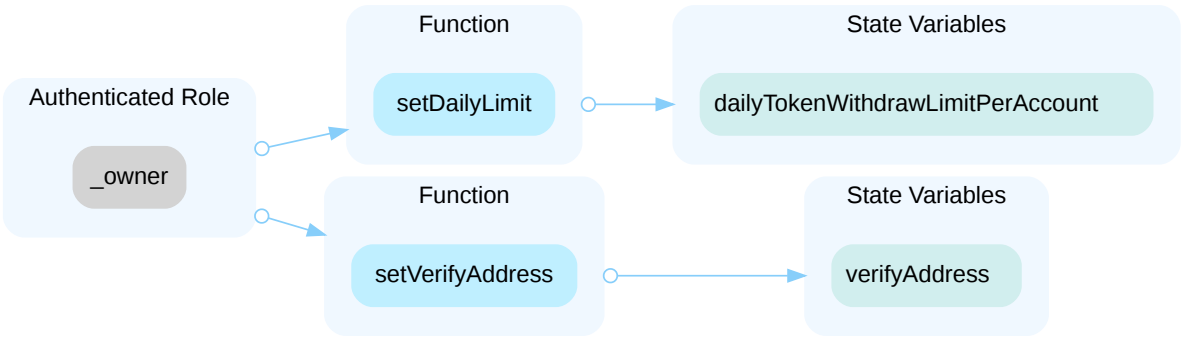
GLOBAL-01 | CENTRALIZATION RELATED RISKS

Category	Severity	Location	Status
Centralization	**Major**		**Acknowledged**

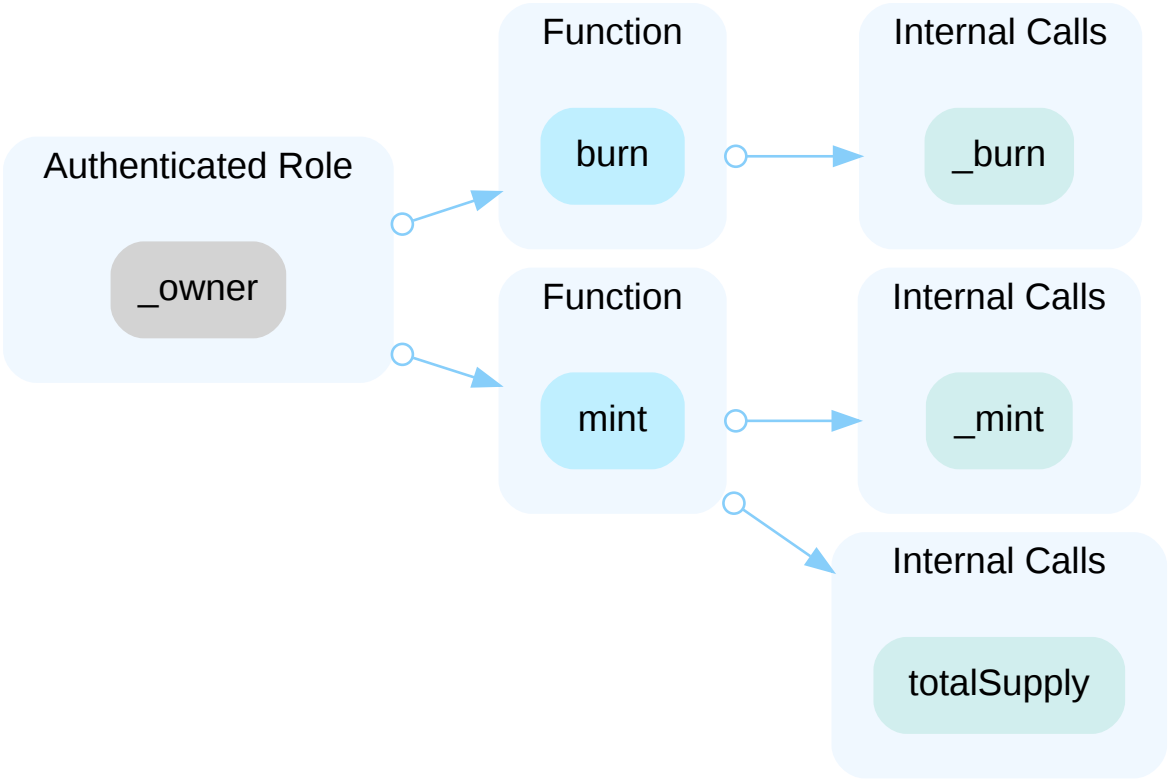
Description

In the contract `Bridge` 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

- Modify the daily withdrawal limit.
- Update the verification address.



In the contract `Dypius` 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 mint tokens to arbitrary addresses. The owner of the contract is currently set to the bridge contract, and the token mint privilege is associated with the bridge's "verifyAddress".



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

The DYP team provided the deployed token and bridge contract on the Binance Smart Chain and the Avalanche blockchain. The 'owner' of the token contract has been set to the bridge contract and cannot be modified in the future. The token minting privilege is associated with the bridge's "verifyAddress". The DYP team acknowledged the finding and has made efforts to reduce the centralization risk associated with the project. However, considering the project is a crypto bridge, and because smart contracts can't communicate across blockchains, they must work with an off-chain centralized component. Thus, it's not possible to completely eliminate the centralization risks.

BSC side:

- Token: https://bscscan.com/address/0x1a3264f2e7b1cfc6220ec9348d33ccf02af7aaa4#code
- Bridge: https://bscscan.com/address/0x9a51ff1005c6825f15696ce5d96783f24e58af89#code

AVAX side:

- Token: https://snowtrace.io/address/0x1a3264f2e7b1cfc6220ec9348d33ccf02af7aaa4#code
- Bridge: https://snowtrace.io/address/0x9a51ff1005c6825f15696ce5d96783f24e58af89#code

CDI-01 | SOLIDITY VERSION NOT RECOMMENDED

Category	Severity	Location	Status
Language Version	Minor	dypius_eth.sol (v2)	Partially Resolved

Description

Solidity frequently releases new compiler versions with improved security features and bug fixes. Using an outdated version prevents access to these enhancements and may leave the smart contract vulnerable to known issues.

dypius_eth.sol:

```
pragma solidity ^0.6.0;
```

Recommendation

It is recommended to deploy with Solidity version ^0.8.0, which offers benefits such as new language features, fewer bugs, and more efficient gas usage, ultimately enhancing code readability and maintainability. Additionally, use a simple pragma version that allows any of these versions. Consider using the latest version of Solidity for testing.

Reference: https://github.com/ethereum/solidity/releases.

Alleviation

The team heeded the advice and resolved the issue in commit 7d6a7489f65ad5593d9facacb6df2dc80fef2a6f for the "dypius.sol" file. The issue still persists in the "dypius_eth.sol" file.

CDU-01 | POTENTIAL CROSS-CHAIN REPLAY ATTACK

Category	Severity	Location	Status
Logical Issue	Minor	bridge_dyp_mint.sol (v1): 640, 678, 688	Resolved

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.

```
640         require(verify(msg.sender, amount, chainId, id, signature),
"invalid signature!");
```

Recommendation

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

Alleviation

The team heeded the advice and resolved the issue in commit 7d6a7489f65ad5593d9facacb6df2dc80fef2a6f.

CDU-02 | INCOMPATIBILITY WITH DEFLATIONARY TOKENS

Category	Severity	Location	Status
Logical Issue	Informational	bridge_dyp_mint.sol (v1): 630, 633	Resolved

Description

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

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

```
630         IERC20(TRUSTED_TOKEN_ADDRESS).safeTransferFrom(msg.sender, address(this
), amount);
```

- Transferring tokens by `amount`.

```
633         dypContract.burn(amount);
```

- The `amount` appears to be used for bookkeeping purposes without compensating the potential transfer fees.
- Note: `burn` is an external function and its behavior wasn't evaluated.

Recommendation

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

Alleviation

The team acknowledged the finding and stated that the contract will not support deflationary tokens.

CDU-04 DELEGATION NOT MOVED ALONG WITH TOKEN-TRANSFER

Category	Severity	Location	Status
Logical Issue	Informational	dypius_eth.sol (7d6a748): 799	Partially Resolved

Description

The voting power of delegation is not moved from one account to another account or `address(0)` along with the `transfer()`、`transferFrom()` and `mint()`.

Recommendation

It's recommended to move delegation along with these functions by calling "_moveDelegates()", if the project design requires such a feature.

Alleviation

The team heeded the advice and resolved the issue in commit 7d6a7489f65ad5593d9facacb6df2dc80fef2a6f for the "dypius.sol" file. The issue still persists in the "dypius_eth.sol" file.

CDU-05 | TOO MANY DIGITS

Category	Severity	Location	Status
Coding Style	Informational	dypius_eth.sol (7d6a748): 802	Resolved

Description

Literals with many digits are difficult to read and review.

Recommendation

We recommend using scientific notation (e.g. `1e18`) or underscores (e.g. `1_000_000`) to improve readability.

Alleviation

The team heeded the advice and resolved the issue in commit e87628d9f7984c0486c3858a1857c9f17355708a

OPTIMIZATIONS | DYPIUS - AUDIT

ID	Title	Category	Severity	Status
CDH-01	Variables That Could Be Declared As Immutable	Gas Optimization	Optimization	Resolved

CDH-01 | VARIABLES THAT COULD BE DECLARED AS IMMUTABLE

Category	Severity	Location	Status
Gas Optimization	Optimization	dypius.sol (v1): 800	Resolved

Description

The linked variables assigned in the constructor can be declared as `immutable`. Immutable state variables can be assigned during contract creation but will remain constant throughout the lifetime of a deployed contract. A big advantage of immutable variables is that reading them is significantly cheaper than reading from regular state variables since they will not be stored in storage.

Recommendation

We recommend declaring these variables as immutable. Please note that the `immutable` keyword only works in Solidity version `v0.6.5` and up.

Alleviation

The team heeded the advice and resolved the issue in commit 9a666716d2cdc535fb90a6f3914896992a54c147.

FORMAL VERIFICATION | DYPIUS - AUDIT

Formal guarantees about the behavior of smart contracts can be obtained by reasoning about properties relating to the entire contract (e.g. contract invariants) or to specific functions of the contract. Once such properties are proven to be valid, they guarantee that the contract behaves as specified by the property. As part of this audit, we applied automated formal verification (symbolic model checking) to prove that well-known functions in the smart contracts adhere to their expected behavior.

Considered Functions And Scope

In the following, we provide a description of the properties that have been used in this audit. They are grouped according to the type of contract they apply to.

Verification of ERC-20 Compliance

We verified properties of the public interface of those token contracts that implement the ERC-20 interface. This covers

- Functions `transfer` and `transferFrom` that are widely used for token transfers,
- functions `approve` and `allowance` that enable the owner of an account to delegate a certain subset of her tokens to another account (i.e. to grant an allowance), and
- the functions `balanceOf` and `totalSupply`, which are verified to correctly reflect the internal state of the contract.

The properties that were considered within the scope of this audit are as follows:

Property Name	Title
erc20-balanceof-change-state	`balanceOf` Does Not Change the Contract's State
erc20-allowance-succeed-always	`allowance` Always Succeeds
erc20-allowance-correct-value	`allowance` Returns Correct Value
erc20-allowance-change-state	`allowance` Does Not Change the Contract's State
erc20-approve-revert-zero	`approve` Prevents Approvals For the Zero Address
erc20-approve-correct-amount	`approve` Updates the Approval Mapping Correctly
erc20-approve-succeed-normal	`approve` Succeeds for Admissible Inputs
erc20-approve-change-state	`approve` Has No Unexpected State Changes
erc20-approve-false	If `approve` Returns `false`, the Contract's State Is Unchanged
erc20-approve-never-return-false	`approve` Never Returns `false`

Property Name	Title
erc20-transfer-revert-zero	`transfer` Prevents Transfers to the Zero Address
erc20-transfer-succeed-normal	`transfer` Succeeds on Admissible Non-self Transfers
erc20-transfer-succeed-self	`transfer` Succeeds on Admissible Self Transfers
erc20-transfer-correct-amount-self	`transfer` Transfers the Correct Amount in Self Transfers
erc20-transfer-correct-amount	`transfer` Transfers the Correct Amount in Non-self Transfers
erc20-transfer-change-state	`transfer` Has No Unexpected State Changes
erc20-transfer-exceed-balance	`transfer` Fails if Requested Amount Exceeds Available Balance
erc20-transfer-recipient-overflow	`transfer` Prevents Overflows in the Recipient's Balance
erc20-transfer-false	If `transfer` Returns `false`, the Contract State Is Not Changed
erc20-transfer-never-return-false	`transfer` Never Returns `false`
erc20-transferfrom-revert-from-zero	`transferFrom` Fails for Transfers From the Zero Address
erc20-transferfrom-revert-to-zero	`transferFrom` Fails for Transfers To the Zero Address
erc20-transferfrom-succeed-normal	`transferFrom` Succeeds on Admissible Non-self Transfers
erc20-transferfrom-succeed-self	`transferFrom` Succeeds on Admissible Self Transfers
erc20-transferfrom-correct-amount	`transferFrom` Transfers the Correct Amount in Non-self Transfers
erc20-transferfrom-correct-amount-self	`transferFrom` Performs Self Transfers Correctly
erc20-transferfrom-correct-allowance	`transferFrom` Updated the Allowance Correctly
erc20-transferfrom-change-state	`transferFrom` Has No Unexpected State Changes
erc20-transferfrom-fail-exceed-balance	`transferFrom` Fails if the Requested Amount Exceeds the Available Balance
erc20-transferfrom-fail-exceed-allowance	`transferFrom` Fails if the Requested Amount Exceeds the Available Allowance
erc20-transferfrom-fail-recipient-overflow	`transferFrom` Prevents Overflows in the Recipient's Balance
erc20-transferfrom-false	If `transferFrom` Returns `false`, the Contract's State Is Unchanged

Property Name	Title
erc20-totalsupply-succeed-always	`totalSupply` Always Succeeds
erc20-transferfrom-never-return-false	`transferFrom` Never Returns `false`
erc20-totalsupply-correct-value	`totalSupply` Returns the Value of the Corresponding State Variable
erc20-totalsupply-change-state	`totalSupply` Does Not Change the Contract's State
erc20-balanceof-succeed-always	`balanceOf` Always Succeeds
erc20-balanceof-correct-value	`balanceOf` Returns the Correct Value

Verification Results

For the following contracts, formal verification established that each of the properties that were in scope of this audit (see scope) are valid:

Detailed Results For Contract ERC20 (Contracts/dypius_eth.sol) In Commit 9a666716d2cdc535fb90a6f3914896992a54c147

Verification of ERC-20 Compliance

Detailed Results for Function `balanceOf`

Property Name	Final Result	Remarks
erc20-balanceof-change-state	True	
erc20-balanceof-succeed-always	True	
erc20-balanceof-correct-value	True	

Detailed Results for Function `allowance`

Property Name	Final Result	Remarks
erc20-allowance-succeed-always	True	
erc20-allowance-correct-value	True	
erc20-allowance-change-state	True	

Detailed Results for Function `approve`

Property Name	Final Result	Remarks
erc20-approve-revert-zero	True	
erc20-approve-correct-amount	True	
erc20-approve-succeed-normal	True	
erc20-approve-change-state	True	
erc20-approve-false	True	
erc20-approve-never-return-false	True	

Detailed Results for Function `transfer`

Property Name	Final Result	Remarks
erc20-transfer-revert-zero	True	
erc20-transfer-succeed-normal	True	
erc20-transfer-correct-amount	True	
erc20-transfer-succeed-self	True	
erc20-transfer-change-state	True	
erc20-transfer-correct-amount-self	True	
erc20-transfer-recipient-overflow	True	
erc20-transfer-exceed-balance	True	
erc20-transfer-false	True	
erc20-transfer-never-return-false	True	

Detailed Results for Function `transferFrom`

Property Name	Final Result	Remarks
erc20-transferfrom-revert-to-zero	True	
erc20-transferfrom-revert-from-zero	True	
erc20-transferfrom-succeed-self	True	
erc20-transferfrom-succeed-normal	True	
erc20-transferfrom-correct-amount-self	True	
erc20-transferfrom-correct-amount	True	
erc20-transferfrom-correct-allowance	True	
erc20-transferfrom-fail-exceed-balance	True	
erc20-transferfrom-fail-exceed-allowance	True	
erc20-transferfrom-change-state	True	
erc20-transferfrom-fail-recipient-overflow	True	
erc20-transferfrom-false	True	
erc20-transferfrom-never-return-false	True	

Detailed Results for Function `totalSupply`

Property Name	Final Result	Remarks
erc20-totalsupply-succeed-always	True	
erc20-totalsupply-correct-value	True	
erc20-totalsupply-change-state	True	

Detailed Results For Contract Dypius (Contracts/dypius_eth.sol) In Commit 9a666716d2cdc535fb90a6f3914896992a54c147

Verification of ERC-20 Compliance

Detailed Results for Function `transfer`

Property Name	Final Result	Remarks
erc20-transfer-revert-zero	True	
erc20-transfer-succeed-normal	True	
erc20-transfer-succeed-self	True	
erc20-transfer-correct-amount-self	True	
erc20-transfer-correct-amount	True	
erc20-transfer-change-state	True	
erc20-transfer-exceed-balance	True	
erc20-transfer-recipient-overflow	True	
erc20-transfer-false	True	
erc20-transfer-never-return-false	True	

Detailed Results for Function `transferFrom`

Property Name	Final Result	Remarks
erc20-transferfrom-revert-from-zero	True	
erc20-transferfrom-revert-to-zero	True	
erc20-transferfrom-succeed-normal	True	
erc20-transferfrom-succeed-self	True	
erc20-transferfrom-correct-amount	True	
erc20-transferfrom-correct-amount-self	True	
erc20-transferfrom-correct-allowance	True	
erc20-transferfrom-change-state	True	
erc20-transferfrom-fail-exceed-balance	True	
erc20-transferfrom-fail-exceed-allowance	True	
erc20-transferfrom-fail-recipient-overflow	True	
erc20-transferfrom-false	True	
erc20-transferfrom-never-return-false	True	

Detailed Results for Function `totalSupply`

Property Name	Final Result	Remarks
erc20-totalsupply-succeed-always	True	
erc20-totalsupply-correct-value	True	
erc20-totalsupply-change-state	True	

Detailed Results for Function `balanceOf`

Property Name	Final Result	Remarks
erc20-balanceof-succeed-always	True	
erc20-balanceof-correct-value	True	
erc20-balanceof-change-state	True	

Detailed Results for Function `allowance`

Property Name	Final Result	Remarks
erc20-allowance-succeed-always	True	
erc20-allowance-correct-value	True	
erc20-allowance-change-state	True	

Detailed Results for Function `approve`

Property Name	Final Result	Remarks
erc20-approve-revert-zero	True	
erc20-approve-succeed-normal	True	
erc20-approve-correct-amount	True	
erc20-approve-false	True	
erc20-approve-change-state	True	
erc20-approve-never-return-false	True	

Detailed Results For Contract Dypius (Contracts/dypius.sol) In Commit 2617f0b85e3eefc01381602c33e073db6b50b437

Verification of ERC-20 Compliance

Detailed Results for Function `approve`

Property Name	Final Result	Remarks
erc20-approve-change-state	True	
erc20-approve-false	True	
erc20-approve-never-return-false	True	
erc20-approve-revert-zero	True	
erc20-approve-correct-amount	True	
erc20-approve-succeed-normal	True	

Detailed Results for Function `transfer`

Property Name	Final Result	Remarks
erc20-transfer-revert-zero	True	
erc20-transfer-succeed-normal	True	
erc20-transfer-succeed-self	True	
erc20-transfer-correct-amount	True	
erc20-transfer-correct-amount-self	True	
erc20-transfer-change-state	True	
erc20-transfer-exceed-balance	True	
erc20-transfer-false	True	
erc20-transfer-recipient-overflow	True	
erc20-transfer-never-return-false	True	

Detailed Results for Function `transferFrom`

Property Name	Final Result	Remarks
erc20-transferfrom-revert-from-zero	True	
erc20-transferfrom-revert-to-zero	True	
erc20-transferfrom-succeed-normal	True	
erc20-transferfrom-succeed-self	True	
erc20-transferfrom-correct-amount	True	
erc20-transferfrom-correct-amount-self	True	
erc20-transferfrom-correct-allowance	True	
erc20-transferfrom-change-state	True	
erc20-transferfrom-fail-exceed-balance	True	
erc20-transferfrom-fail-exceed-allowance	True	
erc20-transferfrom-fail-recipient-overflow	True	
erc20-transferfrom-false	True	
erc20-transferfrom-never-return-false	True	

Detailed Results for Function `totalSupply`

Property Name	Final Result	Remarks
erc20-totalsupply-succeed-always	True	
erc20-totalsupply-correct-value	True	
erc20-totalsupply-change-state	True	

Detailed Results for Function `balanceOf`

Property Name	Final Result	Remarks
erc20-balanceof-succeed-always	True	
erc20-balanceof-correct-value	True	
erc20-balanceof-change-state	True	

Detailed Results for Function `allowance`

Property Name	Final Result	Remarks
erc20-allowance-succeed-always	True	
erc20-allowance-correct-value	True	
erc20-allowance-change-state	True	

Detailed Results For Contract ERC20 (Contracts/dypius.sol) In Commit 2617f0b85e3eefc01381602c33e073db6b50b437

Verification of ERC-20 Compliance

Detailed Results for Function `transfer`

Property Name	Final Result	Remarks
erc20-transfer-revert-zero	True	
erc20-transfer-correct-amount	True	
erc20-transfer-succeed-normal	True	
erc20-transfer-succeed-self	True	
erc20-transfer-correct-amount-self	True	
erc20-transfer-change-state	True	
erc20-transfer-exceed-balance	True	
erc20-transfer-recipient-overflow	True	
erc20-transfer-false	True	
erc20-transfer-never-return-false	True	

Detailed Results for Function `transferFrom`

Property Name	Final Result	Remarks
erc20-transferfrom-revert-from-zero	True	
erc20-transferfrom-revert-to-zero	True	
erc20-transferfrom-succeed-normal	True	
erc20-transferfrom-correct-amount-self	True	
erc20-transferfrom-succeed-self	True	
erc20-transferfrom-correct-amount	True	
erc20-transferfrom-correct-allowance	True	
erc20-transferfrom-change-state	True	
erc20-transferfrom-fail-exceed-balance	True	
erc20-transferfrom-fail-exceed-allowance	True	
erc20-transferfrom-fail-recipient-overflow	True	
erc20-transferfrom-false	True	
erc20-transferfrom-never-return-false	True	

Detailed Results for Function `totalSupply`

Property Name	Final Result	Remarks
erc20-totalsupply-succeed-always	True	
erc20-totalsupply-correct-value	True	
erc20-totalsupply-change-state	True	

Detailed Results for Function `balanceOf`

Property Name	Final Result	Remarks
erc20-balanceof-correct-value	True	
erc20-balanceof-succeed-always	True	
erc20-balanceof-change-state	True	

Detailed Results for Function `allowance`

Property Name	Final Result	Remarks
erc20-allowance-succeed-always	True	
erc20-allowance-correct-value	True	
erc20-allowance-change-state	True	

Detailed Results for Function `approve`

Property Name	Final Result	Remarks
erc20-approve-revert-zero	True	
erc20-approve-succeed-normal	True	
erc20-approve-correct-amount	True	
erc20-approve-change-state	True	
erc20-approve-false	True	
erc20-approve-never-return-false	True	

APPENDIX | DYPIUS - AUDIT

Finding Categories

Categories	Description
Gas Optimization	Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.
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.
Language Version	Language Version findings indicate that the code uses certain compiler versions or language features with known security issues.
Logical Issue	Logical Issue findings indicate general implementation issues related to the program logic.
Centralization	Centralization findings detail the design choices of designating privileged roles or other centralized controls over the code.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Details on Formal Verification

Some Solidity smart contracts from this project have been formally verified. Each such contract was compiled into a mathematical model which reflects all its possible behaviors with respect to the property. The model takes into account the semantics of the Solidity instructions found in the contract. All verification results that we report are based on that model.

Assumptions and Simplifications

The following assumptions and simplifications apply to our model:

- The contract's state variables are non-deterministically initialized before invocation of any function. That may lead to false positives. It is, however, a safe over-approximation.
- The verification engine reasons about unbounded integers. Machine arithmetic is modeled using modular arithmetic based on the bit-width of the underlying numeric Solidity type. This ensures that over- and underflow characteristics are faithfully represented.
- Certain low-level calls and inline assembly are not supported and may lead to a contract not being formally verified.

- We model the semantics of the Solidity source code and not the semantics of the EVM bytecode in a compiled contract.

Formalism for Property Specification

All properties are expressed in CertiK Specification Language that is a specification language derived from Java Modeling Language.

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.