Quantstamp

Sophon - Tokens & BridgeHubWrapper

Executive Summary

This audit report was prepared by Quantstamp, the leader in blockchain security.

Type	Token, NFT, Delegation, and Bridging Wrapper
Timeline	2024-10-14 through 2024-10-19
Language	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	None
Source Code	• sophon-org/support-contracts #b83ee93
Auditors	• Rabib Islam Senior Auditing Engineer • Roman Rohleder Senior Auditing Engineer • Darren Jensen Auditing Engineer

Documentation quality	Medium
Test quality	High
Total Findings	7 Fixed: 3 Acknowledged: 3 Mitigated: 1
High severity findings	0
Medium severity findings	0
Low severity findings	3 Fixed: 1 Acknowledged: 2
Undetermined severity findings	2 Fixed: 1 Mitigated: 1
Informational findings	2 Fixed: 1 Acknowledged: 1

Summary of Findings

The codebase being audited consists of multiple components:

1. `SophonToken`, which is an ERC20 contract;
2. `GuardianNFT`, which is an ERC721 contract;
3. `GuardianDelegation`, which defines logic for delegating the NFTs;
4. `BridgeHubWrapper`, which interacts with zkSync's `Bridgehub` contract to facilitate cross-chain transactions.

The latter three contracts are upgradeable and their proxy contracts are in scope. Particularly noteworthy, however, is the atypical setup: the proxy contracts do not follow ERC1967 and therefore the storage slots particular to the proxy contracts had to be taken into account when defining the storage layout for the implementation contracts. This may lead to issues regarding maintenance in the future, and we have therefore written a couple issues regarding this setup.

Proxy setup aside, the contracts appear to be relatively secure, and no high- or medium-severity issues were found.

The test suite is of high quality. However, more documentation should be included to provide high-level context regarding the overall functionality of the codebase.

Update: Issues were either fixed, acknowledged, or mitigated. The test suite was slightly augmented.

ID	DESCRIPTION	SEVERITY	STATUS
SOP-1	**Base Implementation Contract Storage Does Not Contain Gaps**	Low	Acknowledged
SOP-2	**Proxy's Implementation Address Stored in Default Storage Slot**	Low	Fixed
SOP-3	**Attacker Can Drain `BridgeHubWrapper` of Donated Sophon Tokens**	Low	Acknowledged
SOP-4	**The `requestL2TransactionDirect()` Function Accepts and Forwards Ether to `Bridgehub` Which May Revert**	Informational	Fixed

ID	DESCRIPTION	SEVERITY	STATUS
SOP-5	`BridgeHubWrapper.request*` Functions May Revert if `Bridgehub` is Currently Paused	Informational	Acknowledged
SOP-6	Reentrancy	Undetermined	Fixed
SOP-7	Arbitrary Code Execution Due to Missing Input Validation	Undetermined	Mitigated

Assessment Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

Disclaimer

Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

1. Code review that includes the following
    1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
    2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    3. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
    1. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
    2. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Scope

A number of contracts were specified by the client as in scope for this audit.

Files Included

- `contracts/paymaster-on-l1/BridgeHubWrapper.sol`
- `contracts/paymaster-on-l1/BridgeHubWrapperProxy.sol`
- `contracts/proxies/UpgradeableAccessControl.sol`
- `contracts/tokens/SophonToken.sol`
- `contracts/tokens/GuardianNFT.sol`
- `contracts/tokens/GuardianNFTProxy.sol`
- `contracts/tokens/delegation/GuardianDelegation.sol`
- `contracts/tokens/delegation/GuardianDelegationProxy.sol`

Operational Considerations

1. The contracts being audited are designed to be used in proxy patterns that enable upgrading the contracts; as a result, it is possible for the owner of a proxy to change the logic of the contracts' implementations unilaterally and with impunity, potentially introducing vulnerabilities in the process. **The impact of such changes are outside the scope of this audit.**
2. An admin can set the mapping for `maxReceivedAllowed` by `DelegationType` using the `setMaxReceivedAllowed()` function. However, it is still possible for a receiver to have received more than the `maxReceivedAllowed` if the admin adjusts this to a lower level *after* the receiver has received the delegated NFTs.
3. Token transfers of `GuardianNFT` are blocked within the first 365 days after minting is enabled, save for transfers by the `DEFAULT_ADMIN_ROLE`.
4. NFT holders may delegate part of their balance to "light nodes" or "validators". However, there are no address restrictions, allowing one to delegate to arbitrary addresses, including to oneself.
5. The `BridgeHubWrapper` contract grants unlimited approvals to (out-of-scope) addresses enabled via `toggleSharedBridge()` for the Sophon token. This makes the contract susceptible to be drained in case any of these addresses become compromised.

Key Actors And Their Capabilities

The privileged roles of all contracts, with the exception of `SophonToken`, have extensive privileges that may directly or indirectly lead to loss of funds or the denial of service to the corresponding contracts. We therefore strongly recommend protecting these roles i.e. through the use of multi-sigs to mitigate the potential of abuse or malicious take over.

1. `SophonToken`:
    1. `owner`:
        1. `renounceOwnership()`: Renounce ownership **and thereby prevent any calls to the following listed functions**.
        2. `transferOwnership()`: Transfer ownership in a 2-step process.
        3. `rescue()`: Transfer out any mistakenly sent ERC20 tokens. **But more generally, execute arbitrary code on any contract `token` implementing a `safeTransfer()`-called function.**
2. `BridgeHubWrapper`:
    1. `owner`:
        1. `renounceOwnership()`: See above.
        2. `transferOwnership()`: See above.
        3. `replaceImplementation()`: Update the proxy implementation. **This could lead to unexpected code changes, which are not in scope of this audit!**
        4. `initialize()`: Change the max fee and refund recipient without emitting events. **This function can be called multiple times, contrary to its name.**
        5. `toggleSharedBridge()`: Sets/Unsets unlimitted Sophon token approvals for an arbitrary address. **This could lead to a total loss of Sophon tokens for this contract in case of a compromise!**
        6. `setMaxFee()`: Arbitrarily change the max fee value.
        7. `setRefundRecipient()`: Change the refund recipient address.
        8. `rescue()`: See above. **In particular, this could be used to drain all Sophon tokens from this contract!**
        9. `rescueEth()`: Extract any accidentally send Eth from the contract.
3. `BridgeHubWrapperProxy`:
    1. `owner`:
        1. `renounceOwnership()`: See above.
        2. `transferOwnership()`: See above.
        3. `replaceImplementation()`: See above.
4. `GuardianNFT`:
    1. `DEFAULT_ADMIN_ROLE`:
        1. `renounceRole()`: Renounce the role after a set timer **and therefore prevent any future calls to the following listed functions**.
        2. `changeDefaultAdminDelay()`: Change the timeout schedule.
        3. `rollbackDefaultAdminDelay()`: Reset the timeout schedule.
        4. `grantRole()`: Grant another address a role, other than `DEFAULT_ADMIN_ROLE`.
        5. `revokeRole()`: Revoke the role from another address, other than `DEFAULT_ADMIN_ROLE`.
        6. `replaceImplementation()`: See above.
        7. `initialize()`: **This function can be called multiple times, contrary to its name.**
        8. `setBaseURI()`: Change the base URI string.
        9. `startMinting()`: Enable minting.
        10. `pauseMinting()` / `unpauseMinting()`: Pause/Unpause minting once enabled.
        11. `rescue()`: See above.
        12. `rescueEth()`: See above.
        13. `setDelegationManager()`: Set/Change the delegation manager contract address.
        14. Transfer tokens within the first `365` days after minting, which is otherwise prohibited.
    2. `WHITELIST_MANAGER_ROLE`:
        1. `increaseWhitelist()` / `decreaseWhitelist()`: Increase/Decrease the amount an address may mint.
5. `GuardianNFTProxy`:
    1. `DEFAULT_ADMIN_ROLE`:
        1. `changeDefaultAdminDelay()`: See above.
        2. `rollbackDefaultAdminDelay()`: See above.
        3. `grantRole()`: See above.
        4. `revokeRole()`: See above.

5. `renounceRole()` : See above.
6. `replaceImplementation()` : See above.

6. `GuardianDelegation` :
    1. `DEFAULT_ADMIN_ROLE` :
        1. `changeDefaultAdminDelay()` : See above.
        2. `rollbackDefaultAdminDelay()` : See above.
        3. `grantRole()` : See above.
        4. `revokeRole()` : See above.
        5. `renounceRole()` : See above.
        6. `replaceImplementation()` : See above.
        7. `setMaxReceivedAllowed()` : Set/Change the cap on delegations per type.
        8. `rescue()` : See above.
        9. `rescueEth()` : See above.
7. `GuardianDelegationProxy` :
    1. `DEFAULT_ADMIN_ROLE` :
        1. `changeDefaultAdminDelay()` : See above.
        2. `rollbackDefaultAdminDelay()` : See above.
        3. `grantRole()` : See above.
        4. `revokeRole()` : See above.
        5. `renounceRole()` : See above.
        6. `replaceImplementation()` : See above.

Findings

SOP-1
Base Implementation Contract Storage Does Not Contain Gaps

Low | Acknowledged

Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
No change. We have used these proxy patterns many times in the past and take great care to avoid
storage collisions during upgrades.
```

File(s) affected: `BridgeHubWrapper.sol` , `GuardianNFT.sol` , `GuardianDelegation.sol`

Description: The `BridgeHubWrapper` , `GuardianDelegation` , and `GuardianNFT` contracts are implementation contracts designed to be used in a proxy pattern. These contracts are derived from other contracts that in part define some of their storage slots. When upgrading contracts, it is important that the storage layout of the new implementation aligns with the old implementation. Failure to accomplish this may result in storage collisions, ultimately causing contracts to malfunction.

The above contracts derive their storage layouts from base contracts, namely `Upgradeable2Step` , `UpgradeableAccessControl` , `GuardianNFTState` , and `GuardianDelegationState` . It is typical in cases where implementation contracts are derived from base contracts that the base contracts contain "gap" variables that are intended to remain empty such that if these base contracts are ever upgraded to use extra variables, then these gap variables will be replaced so as to avoid having new variables collide in storage with variables from other base contracts.

Unfortunately, in the current situation, if `UpgradeableAccessControl` is upgraded to include a new variable, this new variable would likely collide with a variable in `GuardianDelegationState` or `GuardianNFTState` (except in the special case where the new variable is given a custom slot).

Recommendation: When using deriving upgradeable contracts from base contracts, ensure the base contracts have gap variables in order to prevent storage collisions when upgrading.

SOP-2
Proxy's Implementation Address Stored in Default Storage Slot

Low | Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `9b7a9fab01c57dd0dbc33812a829777269fd1a25` .

File(s) affected: `UpgradeableAccessControl.sol` , `GuardianDelegation.sol` , `GuardianNFT.sol` , `GuardianDelegationProxy.sol` , `GuardianNFTProxy.sol`

Description: The `UpgradeableAccessControl` contract defines a storage slot for the implementation contract address, `implementation`, of a proxy pattern setup. This contract is inherited by `GuardianDelegation`, `GuardianNFT`, `GuardianDelegationProxy`, and `GuardianNFTProxy` such that the first storage slot in each contract is occupied by `implementation`. In this structure, `GuardianDelegationProxy` is the proxy contract for `GuardianDelegation`, and each of the two share the same storage slot layout. The same relation holds between `GuardianNFTProxy` and `GuardianNFT`.

However, this setup is idiosyncratic: the implementation contracts will be set up so as to never access the first storage slot. This poses a problem: every time the implementation contract is upgraded, the new implementation contract must avoid writing over the first storage slot. Therefore, although in each case as currently established there is no occurrence of a storage collision, upgrades introduce this as a possibility: if a new implementation contract fails to inherit `UpgradeableAccessControl`, it may cause improper writes to the slot used for the `implementation` address.

By contrast, proxies often assign admin and implementation contract address variables to specific slots so as to avoid storage collisions with the implementation contract.

The same issue affects `BridgeHubWrapper` and `BridgeHubWrapperProxy` via the base contract `Upgradeable2Step`.

Recommendation: It may be worth considering storing the implementation address in a special slot, akin to the procedure used in ERC-1967. This would avoid accidentally upgrading to a contract that inappropriately accesses the slot that stores the address of the implementation contract.

SOP-3
Attacker Can Drain `BridgeHubWrapper` of Donated Sophon Tokens

• Low ⓘ Acknowledged

Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
Acknowledged but not fixing. There isn't a feasible way to predict L2 gas costs from L1 without an
oracle. We will load the wrapper with a sufficiently large amount of SOPH to make draining the contract
very expensive. We will monitor and periodically transfer more SOPH to the contract. Also, fees that
arrive on L2 are held by addresses we control, allowing us to claim the SOPH that's accumulated there
once a large amount has gathered. An attacker can grief the contract at their expense but cannot
economically profit from this attack.
```

File(s) affected: `BridgeHubWrapper.sol`

Description: An attacker can repeatedly call the `requestL2TransactionDirect()` function using the request parameters of `_request.mintValue == maxFee` and `_request.l2Value = 0` thereby avoiding any transfer of their own Sophon tokens into the `BridgeHubWrapper` contract. This would result in eventually depleting the Sophon balance in the `BridgeHubWrapper` contract since each call would send `maxFee` Sophon tokens to the `BridgeHub` contract.

Recommendation: Consider preventing unnecessary transfers of the donated Sophon tokens from the `BridgeHubWrapper` contract to the `BridgeHub` contract.

SOP-4
The `requestL2TransactionDirect()` Function Accepts and Forwards Ether to `Bridgehub` Which May Revert

• Informational ⓘ Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `a87ff3594a39b13926c046604cc387ded6182069`.

File(s) affected: `BridgeHubWrapper.sol`

Description: Users can directly bridge their Sophon tokens via the `BridgeHubWrapper` contract by calling the `requestL2TransactionDirect()` function. However, if the user sends any Ether then the call that is forwarded to the `Bridgehub` contract may revert with *"Bridgehub: non-eth bridge with msg.value"* if the registered base token (in the `Bridgehub` contract) for the target chain is not Ether.

Recommendation: Consider only forwarding `msg.value` to the `Bridgehub` contract when the registered base token (in the `Bridgehub` contract) for the target chain is Ether.

SOP-5

`BridgeHubWrapper.request*` Functions May Revert if `Bridgehub` is Currently Paused

Informational | Acknowledged

> **Update**
> Marked as "Acknowledged" by the client.
> The client provided the following explanation:
>
> `No change. Pauses on the BridgeHub should be very rare.`

File(s) affected: `BridgeHubWrapper.sol`

Description: Users can bridge their Sophon tokens via the `BridgeHubWrapper` contract by calling the `requestL2TransactionDirect()` or `requestL2TransactionTwoBridges()` functions. However, the `Bridgehub` contract may be in a paused state at the time the function is called on the `BridgeHubWrapper` contract causing the transaction to revert.

Recommendation: Consider checking the paused status of the `Bridgehub` contract at the start of the function and revert early if the `Bridgehub` contract is currently paused.

SOP-6 Reentrancy

Undetermined | Fixed

> **Update**
> Marked as "Fixed" by the client.
> Addressed in: `5073b1f646dd584b5211d84105424aa8f96234e0` .

File(s) affected: `BridgeHubWrapper.sol`

Related Issue(s): SWC-107

Description: A reentrancy vulnerability is a scenario where an attacker can repeatedly call a function from itself, unexpectedly leading to potentially disastrous results. Here's a basic example representing the very attack which impacted The DAO in 2016:

```
    ```
    function withdraw_with_reentrancy(uint256 _amount) public {
        msg.sender.call.value(_amount)();   // ATTACKER CAN CALL AGAIN BEFORE
                                            // FUNCTION TERMINATES because `call`
                                            // allows msg.sender to execute any code using fallback
function
        balances[msg.sender] -= _amount;   // Balance only updated AFTER funds withdrawn
    }
    ```
```

In particular, the `BridgeHubWrapper.requestL2TransactionTwoBridges()` function is prone to re-entrancy.

Recommendation: We recommend using reentrancy guards and following the pattern checks-effects-interactions to prevent reentrancy.

SOP-7
Arbitrary Code Execution Due to Missing Input Validation

Undetermined | Mitigated

> **Update**
> Marked as "Mitigated" by the client.
> Addressed in: `5073b1f646dd584b5211d84105424aa8f96234e0` .
> The client provided the following explanation:
>
> `No change, but mitigated since we added reentrancy protection (SOP-6)`

File(s) affected: `BridgeHubWrapper.sol`

Description: In function `BridgeHubWrapper.requestL2TransactionTwoBridges()` address `_l1Token` from `_request.secondBridgeCalldata` may point to an arbitrary address leading to arbitrary code execution. In particular, it may trigger the aforementioned reentrancy.

This issue is related to SOP-6.

Note: While out-of-scope for this audit, the custom code could be re-triggered within `L1SharedBridge.bridgeHubDeposit()` → `_depositFunds()`, potentially with different logic by checking `msg.sender` to be the shared bridge contract.

Exploit Scenario:

1. Attacker Alice deploys a contract C, exposing a function `transferFrom(address, address, uint256)` that may contain arbitrary code, including reentrancy to `BridgeHubWrapper.requestL2TransactionTwoBridges()`.
2. Alice calls `BridgeHubWrapper.requestL2TransactionTwoBridges()` with the address of contract C encoded as `_l1Token` in `_request.secondBridgeCalldata`.
3. During execution of `BridgeHubWrapper.requestL2TransactionTwoBridges()` a call is made to `C.transferFrom()`.
4. Arbitrary code is executed within `C.transferFrom()`.
5. A reentrancy is triggered back to `BridgeHubWrapper.requestL2TransactionTwoBridges()` with modified parameters.

Recommendation: We recommend managing a whitelist for the decoded `_l1Token` parameter, to mitigate arbitrary code execution.

Auditor Suggestions

S1 Admin Cannot Prevent NFT Delegation for a Specific Delegation Type

Fixed

Update
Marked as "Fixed" by the client.
Addressed in: `c06eaee301c4847b5923314e23497fae52c11c43`.

File(s) affected: `GuardianDelegation.sol`

Description: Holders of `GuardianNFT` can delegate their NFTs to either a `Validator` or a `LightNode` based on the options in the `DelegationType` enum.

However, while it is possible for an admin to set `maxReceivedAllowed` to restrict the amount that a particular `DelegationType` can receive, it is not possible to set this restriction so as to prevent the user from delegating NFTs since setting the `maxReceivedAllowed` to zero implies allowing unlimited delegations.

Recommendation: Consider adding the option to set `maxReceivedAllowed` in such a way as to prevent users from delegating to a specific `DelegationType`.

S2 `maxReceivedAllowed` Defaults to Allowing Unlimited Delegations

Mitigated

Update
Marked as "Mitigated" by the client.
Addressed in: `c06eaee301c4847b5923314e23497fae52c11c43`.
The client provided the following explanation:

```
No change, but mitigated with the fix for S1
```

File(s) affected: `GuardianDelegation.sol`

Description: Holders of `GuardianNFT` can delegate their NFTs where the number of delegations can be restricted by `maxReceivedAllowed`. However, when the contracts are initially deployed the `maxReceivedAllowed` will default to zero meaning there will initially be no restriction on the amount of NFTs a user can delegate.

Recommendation: Consider setting the `maxReceivedAllowed` to a value that is not the maximum as a default during contract deployment.

S3 Application Monitoring Can Be Improved by Emitting More Events

Fixed

Update
Marked as "Fixed" by the client.

File(s) affected: `BridgeHubWrapper.sol`, `GuardianNFT.sol`, `GuardianDelegation.sol`

Description: In order to validate the proper deployment and initialization of the contracts, it is a good practice to emit events. Additionally, any important state transitions can be logged, which is beneficial for monitoring the contract, and also tracking eventual bugs or hacks. Below we present a non-exhaustive list of events that could be emitted to improve application management:

1. In `GuardianNFT.batchMint()` emit an event for `msg.sender` and `totalQuantity` params.
2. In `GuardianNFT.setBaseURI()` emit an event for `baseURI` param.
3. In `GuardianNFT.startMinting()` emit a generic event to indicate that minting has started.
4. In `GuardianNFT.setDelegationManager()` emit an event for the `delegationManager_` address param.
5. In `GuardianNFT.increaseWhitelist()` emit a generic event to indicate that the whitelist has been increased and by how much.
6. In `GuardianNFT.decreaseWhitelist()` emit a generic event to indicate that the whitelist has been decreased and by how much.
7. In `GuardianDelegation.setMaxReceivedAllowed()` emit an event for `delegationType` and `maxReceivedAllowed_` params.
8. In `BridgeHubWrapper.initialize()` emit an event for the `maxFee` and `refundRecipient` address params.

Recommendation: Consider emitting the events mentioned.

S4 Missing Input Validation

Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `ee48852b7f0f99d4f694a52fad4ae136327bfb26`.

We note however that there is no upper limit for `addedCounts`.

File(s) affected: `BridgeHubWrapper.sol`, `BridgeHubWrapperProxy.sol`, `GuardianNFT.sol`

Related Issue(s): SWC-123

Description: It is important to validate inputs, even if they only come from trusted addresses, to avoid human error.

1. In `BridgeHubWrapper.initialize()` validate `refundRecipient_` address is not a zero address.
2. In `BridgeHubWrapper.initialize()`, `maxFee_` is not checked to be nonzero or otherwise within reasonable bounds.
3. In `BridgeHubWrapper.setRefundRecipient()` validate the `refundRecipient_` address is not a zero address.
4. In `BridgeHubWrapperProxy.constructor` validate the `impl_` address is not a zero address.
5. In `GuardianNFT.increaseWhitelist()` it is missing validation for a user address in the `users` array.
6. In `GuardianNFT.setDelegationManager()` it is missing validation for `delegationManager_` address is not zero and implements the `IGuardianDelegation` interface.
7. In `GuardianNFT.increaseWhitelist()` it is missing validation for an upper limit on `addedCounts`.

Recommendation: We recommend adding the relevant checks.

S5 Miscellaneous Code Improvements

Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `9c031865f9bfeb70737117c677000c4c706e8527`.
The client provided the following explanation:

```
fixed (most of the suggestions):
```

We note the exceptions: implementation of the interface, removal of errors, and using require statements with custom errors.

File(s) affected: `GuardianNFT.sol`, `GuardianDelegation.sol`, `GuardianDelegationState.sol`, `BridgeHubWrapper.sol`

Description: We note a number of aspects of the code that can be improved.

1. In the `GuardianDelegationState` contract the `DelegationType` enum and `Delegation` Struct could be moved to the `IGuardianDelegation` interface.
2. The `GuardianDelegation` contract can implement the `IGuardianDelegation` interface.
3. In the `GuardianDelegationState` contract the `nonces` mapping uses `uint` which should be changed to `uint256` for consistency.
4. In `GuardianNFT` the `Unauthorized` & `LockedForDelegation` errors are unused.
5. Since Solidity `0.8.26` it is possible to use custom errors with `require` statements instead of the "if ... revert" pattern.

Recommendation: Consider addressing the aspects that can be improved.

S6 Ownership Can Be Renounced

Acknowledged

Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
No change, intended behavior
```

File(s) affected: `Upgradeable2Step.sol`, `SophonToken.sol`, `GuardianNFT.sol`, `GuardianDelegation.sol`, `BridgeHubWrapper.sol`, `GuardianNFTProxy.sol`, `GuardianDelegationProxy.sol`, `BridgeHubWrapperProxy.sol`

Description: If the owner renounces their ownership, all ownable contracts will be left without an owner. Consequently, any function guarded by the `onlyOwner` modifier will no longer be able to be executed.

The following contracts may have their ownership renounced:

1. `GuardianNFT`
2. `GuardianNFTProxy`
3. `SophonToken`
4. `GuardianDelegation`
5. `GuardianDelegationProxy`
6. `BridgeHubWrapper`
7. `BridgeHubWrapperProxy`

Recommendation: Confirm that this is the intended behavior. If not, override and disable the `renounceOwnership()` function in the affected contracts.

S7 Unlicensed Contracts

Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `c2b7d50827498a08dadfa06f5503af0c236f3c05`.

File(s) affected: `All`

Description: A license, as marked at the top of a contract after `// SPDX-License-Identifier:`, may help control who may copy and modify these contracts and to what extend. A missing license, i.e. as marked via `UNLICENSED`, forfeits any such control and allows for arbitrary modifications and could even be re-licensed with different terms, potentially preventing the original author to perform any changes.

All contracts from this protocol that are in scope are currently marked as `UNLICENSED` and could therefore be subject to above mentioned changes.

Recommendation: We recommend using a proper license before publishing the code to prevent undesired code re-use and modifications.

S8 Magic Constants

Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `4389b426d12e7be991df5dca9fa84e70a2165344`.

File(s) affected: `GuardianNFT.sol`, `BridgeHubWrapper.sol`

Description: To improve readability and lower the risk of introducing errors when making code changes, it is advised to not use magic constants throughout code, but instead declare them once (as constant and commented) and use these constant variables instead. The following instances should therefore be changed accordingly:

1. `GuardianNFT._beforeTokenTransfer()`: `365 days`.
2. `BridgeHubWrapper.getL2Alias()`: `0x1111000000000000000000000000000000001111`.

Recommendation: Consider making the adjustments.

S9 Unnecessary Inheritance

Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `d2103f09ba3f36bbadd17d50cc89b653f6377047`.

File(s) affected: `GuardianNFTState.sol`

Description: `ERC721A` is being inherited unnecessarily since the inherited `ERC721AQueryable` implicitly inherits `ERC721A`.

Recommendation: Remove the unnecessary inheritance.

S10 Multiple Redundant Implementations of Functions

Fixed

> **Update**
> Marked as "Fixed" by the client.
> Addressed in: `e6003fa0f78763d37130e006022ec96c56e142f2`.
> The client provided the following explanation:
>
> ```
> fixed (some of the suggestions)
> ```

File(s) affected: `BridgeHubWrapper.sol`, `SophonToken.sol`, `GuardianDelegation.sol`, `GuardianNFT.sol`, `Proxy2Step.sol`, `ProxyAccessControl.sol`

Description: The following functions are defined multiple times.

1. `rescue()` and `rescueEth()`:
    1. `BridgeHubWrapper.sol`.
    2. `SophonToken.sol`.
    3. `GuardianDelegation.sol`.
    4. `GuardianNFT.sol`.
2. `fallback()` (proxy caller):
    1. `BridgeHubWrapper.sol`.
    2. `Proxy2Step.sol`.
    3. `ProxyAccessControl.sol`.

Recommendation: For improved maintainability we recommend defining above mentioned functions only once, i.e. in a separate `Utils` library, and re-using them.

S11 Unnecessary Use `receive()` and `rescueEth()` Functions

Acknowledged

> **Update**
> Marked as "Acknowledged" by the client.
> The client provided the following explanation:
>
> ```
> No change, leaving these in some these are proxies
> ```

File(s) affected: `BridgeHubgWrappperProxy.sol`, `Proxy2Step.sol`, `ProxyAccessControl.sol`

Description: The following functions contain a payable `receive()` function (and a `rescueEth()` in the implementation contracts), without the need of receiving native token payments:

1. `BridgeHubgWrappperProxy.sol`.
2. `Proxy2Step.sol`.
3. `ProxyAccessControl.sol`.

Recommendation: We recommend removing these functions to prevent unnecessary receipt of native token payments.

Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
- **Informational** – The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.

- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **Acknowledged** – The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Files

- `e68...507 ./contracts/tokens/GuardianNFTState.sol`
- `afc...d12 ./contracts/tokens/SophonToken.sol`
- `43a...093 ./contracts/tokens/GuardianNFTProxy.sol`
- `8f0...ee5 ./contracts/tokens/GuardianNFT.sol`
- `de5...1cc ./contracts/tokens/delegation/IGuardianDelegation.sol`
- `cfe...dfb ./contracts/tokens/delegation/GuardianDelegationState.sol`
- `3df...b3d ./contracts/tokens/delegation/GuardianDelegation.sol`
- `d90...ca9 ./contracts/tokens/delegation/GuardianDelegationProxy.sol`
- `4f0...832 ./contracts/proxies/Upgradeable2Step.sol`
- `522...7e9 ./contracts/proxies/Upgradeable.sol`
- `068...b7e ./contracts/proxies/Proxy2Step.sol`
- `a98...5f1 ./contracts/proxies/Proxy.sol`
- `a97...de0 ./contracts/proxies/ProxyAccessControl.sol`
- `c48...9a9 ./contracts/proxies/UpgradeableAccessControl.sol`
- `4a3...5a0 ./contracts/wsoph/L2WETH.sol`
- `c8d...44a ./contracts/wsoph/interfaces/IL1Bridge.sol`
- `ccb...f93 ./contracts/wsoph/interfaces/IL2Bridge.sol`
- `c7d...501 ./contracts/wsoph/interfaces/IL2StandardToken.sol`
- `553...a03 ./contracts/wsoph/interfaces/IL2WETH.sol`
- `821...dc8 ./contracts/paymaster-on-l1/BridgeHubWrapper.sol`
- `24c...052 ./contracts/paymaster-on-l1/BridgeHubWrapperProxy.sol`
- `3b4...48b ./contracts/mocks/MockERC20.sol`

Tests

- `91d...da7 ./test/GuardianDelegation.t.sol`
- `95d...1e6 ./test/SophonToken.t.sol`
- `16e...cf4 ./test/GuardianDelegation.py`
- `18c...5a6 ./test/L2WETH.t.sol`
- `c40...d8a ./test/GuardianNFT.t.sol`
- `88c...fdd ./test/BridgeHubWrapper.t.sol`
- `0bb...907 ./test/interfaces/EraContractsInterfaces.sol`
- `515...6d3 ./test/harness/GuardianNFTHarness.sol`

Toolset

The notes below outline the setup and steps performed in the process of this audit.

Setup

Tool Setup:

- Slither v0.10.0

Steps taken to run the tools:

1. Install the Slither tool: `pip3 install slither-analyzer`
2. Run Slither from the project directory: `slither .`

Automated Analysis

Slither

No serious issues were found in the Slither results.

Test Suite Results

The test suite is relatively comprehensive and includes tests that interact with contracts deployed on Sepolia via a fork.

Update: Six tests were added for `GuardianNFT`.

```
Ran 7 tests for test/SophonToken.t.sol:SophonTokenTest
[PASS] test_Burn(uint256) (runs: 500, μ: 32130, ~: 31632)
[PASS] test_BurnFrom(uint256) (runs: 500, μ: 45926, ~: 45488)
[PASS] test_CorrectSetUp() (gas: 23533)
[PASS] test_Name() (gas: 10543)
[PASS] test_Rescue() (gas: 1713209)
[PASS] test_Symbol() (gas: 10586)
[PASS] test_Update() (gas: 12514)
Suite result: ok. 7 passed; 0 failed; 0 skipped; finished in 478.45ms (552.15ms CPU time)

Ran 36 tests for test/GuardianDelegation.t.sol:GuardianDelegationTest
[PASS] test_CorrectSetUp() (gas: 46132)
[PASS] test_DelegateBySignature_LightNode() (gas: 304561)
[PASS] test_DelegateBySignature_Validator() (gas: 304529)
[PASS] test_DelegateToLightNode() (gas: 274915)
[PASS] test_DelegateToLightNodes_WithAmount(uint256) (runs: 500, μ: 1161655, ~: 1069484)
[PASS] test_DelegateToLightNodes_WithoutAmount() (gas: 636280)
[PASS] test_DelegateToValidator() (gas: 266416)
[PASS] test_DelegateToValidators_WithAmount(uint256) (runs: 500, μ: 2199997, ~: 2200184)
[PASS] test_DelegateToValidators_WithoutAmount() (gas: 449386)
[PASS] test_Delegate_ExactlyMaxReceivedAllowed() (gas: 381619)
[PASS] test_Delegate_FuzzMaxReceivedAllowed(uint256) (runs: 500, μ: 472527, ~: 473178)
[PASS] test_Delegate_LessThanMaxReceivedAllowed(uint256) (runs: 500, μ: 344190, ~: 347538)
[PASS] test_Delegate_MoreThanMaxReceivedAllowed(uint256) (runs: 500, μ: 386894, ~: 386930)
[PASS] test_Delegate_WithoutMaxAllowed(uint256) (runs: 500, μ: 440002, ~: 442281)
[PASS] test_FuzzRemoveDelegation(uint256) (runs: 500, μ: 288222, ~: 288573)
[PASS] test_GuardianNFT() (gas: 16458)
[PASS] test_RemoveDelegationToAllLightNodes() (gas: 225584)
[PASS] test_RemoveDelegationToAllLightNodes_MultipleAmounts(uint256) (runs: 500, μ: 1009196, ~: 931556)
[PASS] test_RemoveDelegationToAllValidators() (gas: 225640)
[PASS] test_RemoveDelegationToAllValidators_MultipleDelegations(uint256) (runs: 500, μ: 1881102, ~:
1881210)
[PASS] test_RemoveDelegationToLightNode() (gas: 225118)
[PASS] test_RemoveDelegationToValidator() (gas: 225103)
[PASS] test_Rescue() (gas: 1716710)
[PASS] test_RescueEth() (gas: 29201)
[PASS] test_RevertIfCountMismatch_DelegateToLightNodes() (gas: 109693)
[PASS] test_RevertIfCountMismatch_DelegateToValidators() (gas: 109716)
[PASS] test_RevertIfECDSAInvalidSignature_DelegateBySignature() (gas: 31829)
[PASS] test_RevertIfInvalidDelegatee_Delegate() (gas: 17185)
[PASS] test_RevertIfInvalidDelegationType_DelegateBySignature() (gas: 58781)
[PASS] test_RevertIfInvalidNonce_DelegateBySignature() (gas: 23878)
[PASS] test_RevertIfInvalidSignature_DelegateBySignature() (gas: 38331)
[PASS] test_RevertIfNoAdmin_Rescue() (gas: 1511682)
[PASS] test_RevertIfNoAdmin_RescueEth() (gas: 19992)
[PASS] test_RevertIfSignatureExpired_DelegateBySignature() (gas: 21849)
[PASS] test_RevertIfZeroBalance_Delegate() (gas: 30476)
[PASS] test_SetMaxReceivedAllowed(uint256) (runs: 500, μ: 28585, ~: 28643)
Suite result: ok. 36 passed; 0 failed; 0 skipped; finished in 3.79s (11.71s CPU time)

Ran 10 tests for test/L2WETH.t.sol:W2WETHTest
[PASS] test_BridgeBurn() (gas: 17539)
```

```
[PASS] test_BridgeMint() (gas: 17432)
[PASS] test_Deposit(uint256) (runs: 500, μ: 79390, ~: 78777)
[PASS] test_DepositTo(uint256) (runs: 500, μ: 82067, ~: 81454)
[PASS] test_Name() (gas: 18267)
[PASS] test_Symbol() (gas: 18377)
[PASS] test_Withdraw(uint256,uint256) (runs: 500, μ: 95658, ~: 96451)
[PASS] test_WithdrawTo(uint256,uint256) (runs: 500, μ: 123130, ~: 124184)
[PASS] test_l1Address() (gas: 16627)
[PASS] test_l2Bridge() (gas: 16605)
Suite result: ok. 10 passed; 0 failed; 0 skipped; finished in 198.11s (1.11s CPU time)

Ran 53 tests for test/GuardianNFT.t.sol:GuardianNFTTest
[PASS] test_AdminCanTransferBeforeTheYear_BeforeTokenTransfers() (gas: 253073)
[PASS] test_BaseURI() (gas: 5096278)
[PASS] test_BatchMint(uint256) (runs: 500, μ: 535457, ~: 566122)
[PASS] test_BatchTransferFrom(uint256) (runs: 500, μ: 1579195, ~: 1625506)
[PASS] test_CorrectSetUp() (gas: 36014)
[PASS] test_DecreaseWhitelist(uint256) (runs: 500, μ: 207230, ~: 228021)
[PASS] test_DecreaseWhitelist_HigherAmounts(uint256) (runs: 500, μ: 180816, ~: 193537)
[PASS] test_IncreaseWhitelist(uint256) (runs: 500, μ: 195918, ~: 209422)
[PASS] test_Mint(uint256) (runs: 500, μ: 695800, ~: 745221)
[PASS] test_Name() (gas: 15634)
[PASS] test_PauseMinting() (gas: 50654)
[PASS] test_ReplaceImplementation() (gas: 5102847)
[PASS] test_Rescue() (gas: 1716602)
[PASS] test_RescueEth() (gas: 29202)
[PASS] test_RevertIfCMintingDisabled_BatchMint() (gas: 22053)
[PASS] test_RevertIfContractIsReceiver_BeforeTokenTransfers() (gas: 197026)
[PASS] test_RevertIfCountMismatch_BatchMint() (gas: 69191)
[PASS] test_RevertIfCountMismatch_DecreaseWhitelist() (gas: 22819)
[PASS] test_RevertIfCountMismatch_IncreaseWhitelist() (gas: 22840)
[PASS] test_RevertIfIncorrectERC721Receiver_SafeBatchTransferFrom(uint256) (runs: 500, μ: 749455, ~:
748373)
[PASS] test_RevertIfLockedForDelegation_BeforeTokenTransfers() (gas: 6434149)
[PASS] test_RevertIfMintingDisabled_Mint() (gas: 19432)
[PASS] test_RevertIfMintingNotStarted_PauseMinting() (gas: 20891)
[PASS] test_RevertIfMintingNotStarted_UnpauseMinting() (gas: 20824)
[PASS] test_RevertIfMintingPaused_PauseMinting() (gas: 50781)
[PASS] test_RevertIfMintingStarted_StartMinting() (gas: 65503)
[PASS] test_RevertIfMintingUnpaused_UnpauseMinting() (gas: 65723)
[PASS] test_RevertIfNoAdmin_PauseMinting() (gas: 20042)
[PASS] test_RevertIfNoAdmin_Rescue() (gas: 1511706)
[PASS] test_RevertIfNoAdmin_RescueEth() (gas: 20056)
[PASS] test_RevertIfNoAdmin_SetDelegationManager() (gas: 22662)
[PASS] test_RevertIfNoAdmin_StartMinting() (gas: 19957)
[PASS] test_RevertIfNoAdmin_UnpauseMinting() (gas: 19997)
[PASS] test_RevertIfNoWhitelistManager_DecreaseWhitelist() (gas: 25042)
[PASS] test_RevertIfNoWhitelistManager_IncreaseWhitelist() (gas: 25085)
[PASS] test_RevertIfNonERC721Receiver_SafeBatchTransferFrom(uint256) (runs: 500, μ: 751128, ~: 750046)
[PASS] test_RevertIfQuantityTooHigh_BatchMint() (gas: 193603)
[PASS] test_RevertIfQuantityTooHigh_BeforeTokenTransfers() (gas: 6501907)
[PASS] test_RevertIfQuantityTooHigh_Mint() (gas: 101986)
[PASS] test_RevertIfRevertsERC721Receiver_SafeBatchTransferFrom(uint256) (runs: 500, μ: 782237, ~:
781155)
[PASS] test_RevertIfTransferNotAllowed_BeforeTokenTransfers() (gas: 204824)
[PASS] test_RevertIfZeroQuantity_BatchMint() (gas: 68812)
[PASS] test_RevertIfZeroQuantity_Mint() (gas: 66041)
[PASS] test_SafeBatchTransferFrom(uint256) (runs: 500, μ: 1581820, ~: 1628131)
[PASS] test_SafeBatchTransferFrom_WithData(uint256) (runs: 500, μ: 806502, ~: 812692)
[PASS] test_SetBaseURI() (gas: 49476)
[PASS] test_SetDelegationManager() (gas: 46907)
[PASS] test_StartMinting() (gas: 67807)
[PASS] test_SupportsInterface() (gas: 16195)
[PASS] test_Symbol() (gas: 15701)
[PASS] test_UnpauseMinting() (gas: 71568)
[PASS] test_UserCanTransferAfterTheYear_BeforeTokenTransfers() (gas: 253197)
[PASS] test_Whitelist() (gas: 50931)
Suite result: ok. 53 passed; 0 failed; 0 skipped; finished in 198.12s (16.62s CPU time)

Ran 30 tests for test/BridgeHubWrapper.t.sol:BridgeHubWrapperTest
[PASS] test_CorrectSetUp() (gas: 38212)
[PASS] test_FallbackFunction_Revert() (gas: 28298)
```

```
[PASS] test_FallbackFunction_Success() (gas: 35238)
[PASS] test_GetL2Alias() (gas: 14828)
[PASS] test_L2Address() (gas: 16387)
[PASS] test_RequestL2TransactionDirect() (gas: 338995)
[PASS] test_Rescue() (gas: 1716430)
[PASS] test_RescueEth() (gas: 28900)
[PASS] test_RevertIfFeeTooHigh_RequestL2TransactionTwoBridges() (gas: 52771)
[PASS] test_RevertIfInvalidChainId_RequestL2TransactionDirect() (gas: 83656)
[PASS] test_RevertIfInvalidChain_RequestL2TransactionTwoBridges() (gas: 46203)
[PASS] test_RevertIfInvalidFeeTooHigh_RequestL2TransactionDirect() (gas: 85778)
[PASS] test_RevertIfInvalidMintValue_RequestL2TransactionDirect() (gas: 83516)
[PASS] test_RevertIfInvalidMintValue_RequestL2TransactionTwoBridges() (gas: 50422)
[PASS] test_RevertIfInvalidSecondBridgeAddress_RequestL2TransactionTwoBridges() (gas: 48244)
[PASS] test_RevertIfInvalidSecondBridgeCalldata_RequestL2TransactionTwoBridges() (gas: 101788)
[PASS] test_RevertIfInvalidSecondBridgeValue_RequestL2TransactionTwoBridges() (gas: 57077)
[PASS] test_RevertIfNoAdmin_Rescue() (gas: 1511169)
[PASS] test_RevertIfNoAdmin_RescueEth() (gas: 19449)
[PASS] test_RevertIfNoOwner_Initialize() (gas: 24443)
[PASS] test_RevertIfNoOwner_SetMaxFee() (gas: 19920)
[PASS] test_RevertIfNoOwner_SetRefundRecipient() (gas: 20182)
[PASS] test_RevertIfNotEnoughFeeToken_RequestL2TransactionDirect() (gas: 185555)
[PASS] test_RevertIfNotEnoughFeeToken_RequestL2TransactionTwoBridges() (gas: 153701)
[PASS] test_SetMaxFee(uint256) (runs: 500, μ: 27293, ~: 27351)
[PASS] test_SetRefundRecipient(address) (runs: 500, μ: 28516, ~: 28516)
[PASS] test_ToggleSharedBridge() (gas: 34565)
[PASS] test_Token_RequestL2TransactionTwoBridges(uint256) (runs: 500, μ: 564397, ~: 563961)
[PASS] test_WithL2Value_RequestL2TransactionTwoBridges(uint256) (runs: 500, μ: 497263, ~: 496636)
[PASS] test_WithValue_RequestL2TransactionTwoBridges(uint256) (runs: 500, μ: 407266, ~: 406830)
Suite result: ok. 30 passed; 0 failed; 0 skipped; finished in 198.12s (576.84s CPU time)

Ran 5 test suites in 198.13s (598.61s CPU time): 136 tests passed, 0 failed, 0 skipped (136 total tests)
```

Code Coverage

Test coverage is good. However, `Upgradeable2Step` lacks coverage, and it is a base contract for `BridgeHubWrapperProxy`.

Update: The quality of coverage is largely the same.

File	% Lines	% Statements	% Branches	% Funcs
contracts/common/Rescuable.sol	100.00% (7/7)	100.00% (9/9)	50.00% (1/2)	100.00% (2/2)
contracts/paymaster-on-l1/BridgeHubWrapper.sol	100.00% (54/54)	100.00% (62/62)	87.18% (34/39)	100.00% (11/11)
contracts/paymaster-on-l1/BridgeHubWrapperProxy.sol	100.00% (4/4)	100.00% (5/5)	60.00% (3/5)	100.00% (1/1)
contracts/proxies/Proxy.sol	0.00% (0/7)	0.00% (0/11)	100.00% (0/0)	0.00% (0/2)
contracts/proxies/Proxy2Step.sol	100.00% (9/9)	100.00% (13/13)	50.00% (1/2)	100.00% (2/2)
contracts/proxies/ProxyAccessControl.sol	100.00% (7/7)	100.00% (11/11)	100.00% (0/0)	100.00% (2/2)
contracts/proxies/Upgradeable.sol	0.00% (0/6)	0.00% (0/6)	100.00% (0/0)	0.00% (0/3)
contracts/proxies/Upgradeable2Step.sol	0.00% (0/21)	0.00% (0/23)	0.00% (0/2)	16.67% (1/6)

File	% Lines	% Statements	% Branches	% Funcs
contracts/proxies/UpgradeableAccessControl.sol	100.00% (11/11)	100.00% (13/13)	60.00% (3/5)	100.00% (4/4)
contracts/tokens/GuardianNFT.sol	98.81% (83/84)	99.17% (120/121)	93.10% (27/29)	100.00% (21/21)
contracts/tokens/GuardianNFTProxy.sol	100.00% (0/0)	100.00% (0/0)	100.00% (0/0)	100.00% (1/1)
contracts/tokens/SophonToken.sol	100.00% (17/17)	100.00% (21/21)	100.00% (1/1)	100.00% (7/7)
contracts/tokens/delegation/GuardianDelegation.sol	99.05% (104/105)	98.48% (130/132)	94.44% (17/18)	91.30% (21/23)
contracts/tokens/delegation/GuardianDelegationProxy.sol	100.00% (0/0)	100.00% (0/0)	100.00% (0/0)	100.00% (1/1)
contracts/wsoph/L2WETH.sol	100.00% (14/14)	100.00% (15/15)	50.00% (1/2)	100.00% (10/10)
Total	89.60% (310/346)	90.27% (399/442)	83.81% (88/105)	87.50% (84/96)

Changelog

- 2024-10-21 - Initial report
- 2024-11-15 - Final report

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp's team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over $200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites&aspo; owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and and may not be represented as such. No third party is entitled to rely on the report in any any way, including for the purpose of making any decisions to buy or sell a product, product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or or any open source or third-party software, code, libraries, materials, or information to, to, called by, referenced by or accessible through the report, its content, or any related related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.