Quantstamp Security Assessment Certificate

July 22nd 2022 — Quantstamp Verified

Meta Studio

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

Executive Summary

Type	ERC20 Token
Auditors	Fatemeh Heidari, Security Auditor Souhail Mssassi, Research Engineer Andy Lin, Senior Auditing Engineer
Timeline	2022-07-18 through 2022-07-22
EVM	Arrow Glacier
Languages	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	None
Documentation Quality	High
Test Quality	Medium

Source Code

Repository	Commit
MetaStudio	034910f
MetaStudio (re-audit)	8d130d4

Total Issues	**5** (4 Resolved)
High Risk Issues	0 (0 Resolved)
Medium Risk Issues	0 (0 Resolved)
Low Risk Issues	**4** (3 Resolved)
Informational Risk Issues	**1** (1 Resolved)
Undetermined Risk Issues	0 (0 Resolved)

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
Low Risk	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.
Unresolved	Acknowledged the existence of the risk, and decided to accept it without engaging in special efforts to control it.
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).
Fixed	Adjusted program implementation, requirements or constraints to eliminate the risk.
Mitigated	Implemented actions to minimize the impact or likelihood of the risk.

Summary of Findings

Meta Studio is an ERC20 token that implements several standards, including ERC165, ERC2771, ERC1363, ERC712, ERC2612, ERC1967, and ERC1822 on top of ERC20. Quantstamp has found five issues in total, which all were of low, informational, and undetermined severity. besides some best practices. Nonetheless, we recommend addressing all points before deploying in production. We strongly recommend adding tests for `ERC2771ContextUpgradeable` to gain coverage higher than 90%.
Update: We reviewed commit 8d130d4 including the changes related to the issues in initial report, and all issues are fixed, acknowledged, or mitigated. About the acknowledged and mitigated issues, the MetaStudio team has the responsibility to manage related problematic situations and communicate about them with the users.

ID	Description	Severity	Status
QSP-1	Useless Reentrancy Protection	Low	Fixed
QSP-2	Admin Role Can Be Renounced	Low	Acknowledged
QSP-3	Unlocked Pragma	Low	Fixed
QSP-4	Allowance Double-Spend Exploit	Low	Mitigated
QSP-5	Upgradable Contracts Are Not Initialized	Informational	Fixed

Quantstamp Audit Breakdown

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

DISCLAIMER:
This report is limited to the following files:

- contracts/ERC1363/ERC1363Upgradeable.sol
- contracts/metatx/ERC2771ContextUpgradeable.sol
- contracts/IERC2771Upgradeable.sol
- contracts/Proxy/import.sol
- contracts/tokens/IPausable.sol
- contracts/tokens/MetaStudioToken.sol

If the final commit hash provided by the client contains features that are not in scope of the audit or a re-audit, those features are excluded from the 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

The Quantstamp auditing process follows a routine series of steps:

1. Code review that includes the following
   i. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   iii. 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:
   i. 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.
   ii. 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, clarify, 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.

Toolset

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

Setup

Tool Setup:

- Slither v0.8.2

Steps taken to run the tools:

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

Findings

QSP-1 Useless Reentrancy Protection

Severity: *Low Risk*

Status: Fixed

File(s) affected: `MetaStudioToken`, `ERC1363Upgradeable`

Description: The function `MetaStudioToken.sol:_beforeTokenTransfer(...)` in L125-131 uses the modifier `nonReentrant` to protect against reentrancy attack. However, the `nonReentrant` modifier implementation from the OpenZeppelin only guards against the life cycle of the function attached with the modifier. In other words, it only protects that reentrancy will fail within the function of `_beforeTokenTransfer(...)`. It will not protect the reentrancy triggered from `IERC1363ReceiverUpgradeable(...).onTransferReceived(...)` and `IERC1363SpenderUpgradeable(...).onApprovalReceived(...)` in the `ERC1363Upgradeable.sol` contract.

Exploit Scenario: Alice calls `ERC1363Upgradeable.sol:transferAndCall` and sends tokens to a malicious contract that will re-enter the token contract.

Recommendation: Commonly, the reentrancy is protected on the integration contract instead of the token contract. Thus, we recommend simply removing the `nonReentrant` modifier and the inheritance of `ReentrancyGuardUpgradeable`. Nonetheless, if the team prefers to embed the protection on the token contract, please explicitly add the `nonReentrant` in the functions of `ERC1363Upgradeable.sol` contracts.

Update: The Meta Studio team removed the `nonReentrant` modifier for `MetaStudioToken.sol:_beforeTokenTransfer` function.

QSP-2 Admin Role Can Be Renounced

Severity: *Low Risk*

Status: Acknowledged

Description: If the `DEFAULT_ADMIN_ROLE` renounces their role, then `MetaStudioToken` will be left without any privileged users. Consequently, functions with access control restricted to `onlyRole(DEFAULT_ADMIN_ROLE)` will no longer be able to be executed.

Recommendation: Double check if this is the intended behavior.

Update: The Meta Studio team stated that they double-checked the issue and it's the intended behavior.

QSP-3 Unlocked Pragma

Severity: *Low Risk*

Status: Fixed

File(s) affected: `MetaStudioToken`, `ERC1363Upgradeable`, `ERC2771ContextUpgradeable`, `IERC2771Upgradeable`, `ERC1363ReceiverMock`, `ERC1363SpenderMock`, `IPausable`

Related Issue(s): SWC-103

Description: Every Solidity file specifies in the header a version number of the format `pragma solidity (^)0.8.*`. The caret (^) before the version number implies an unlocked pragma, meaning that the compiler will use the specified version *and above*, hence the term "unlocked".

Recommendation: For consistency and to prevent unexpected behavior in the future, we recommend to remove the caret to lock the file onto a specific Solidity version.

Update: The version is locked to `0.8.7`.

QSP-4 Allowance Double-Spend Exploit

Severity: *Low Risk*

Status: Mitigated

File(s) affected: `MetaStudioToken`

Description: As it presently is constructed, the contract is vulnerable to the allowance double-spend exploit, as with other ERC20 tokens.

Exploit Scenario:

1. Alice allows Bob to transfer `N` amount of Alice's tokens (`N>0`) by calling the `approve()` method on `Token` smart contract (passing Bob's address and `N` as method arguments)
2. After some time, Alice decides to change from `N` to `M` (`M>0`) the number of Alice's tokens Bob is allowed to transfer, so she calls the `approve()` method again, this time passing Bob's address and `M` as method arguments
3. Bob notices Alice's second transaction before it was mined and quickly sends another transaction that calls the `transferFrom()` method to transfer `N` Alice's tokens somewhere
4. If Bob's transaction will be executed before Alice's transaction, then Bob will successfully transfer `N` Alice's tokens and will gain an ability to transfer another `M` tokens
5. Before Alice notices any irregularities, Bob calls `transferFrom()` method again, this time to transfer `M` Alice's tokens.

Recommendation: The exploit (as described above) is mitigated through use of functions that increase/decrease the allowance relative to its current value, such as `increaseAllowance()` and

`decreaseAllowance()`. Furthermore, we recommend that developers of applications dependent on `approve()` / `transferFrom()` should keep in mind that they have to set allowance to 0 first and verify if it was used before setting the new value.

Update: The issue is mitigated with the `increaseAllowance` and `decreaseAllowance` functions. Still, we recommend adding this to documentation and encouraging users to use the increase/decrease allowance functions instead of simply using the approve function.

QSP-5 Upgradable Contracts Are Not Initialized

Severity: Informational

Status: Fixed

File(s) affected: `MetaStudioToken`

Description: The `ContextUpgradeable`, and `UUPSUpgradeable` contracts are inherited in `MetaStudioToken` contract but never initialized. Although these functions have no implementation all of them are upgradable contracts and might change in future.

Recommendation: We recommend initializing `ContextUpgradeable`, and `UUPSUpgradeable` by calling `__Context_init`, `__UUPSUpgradeable_init` in the `initialize()` function.

Automated Analyses

Slither

Slither reported 30 issues which the most severe one was related to the protecting initialize function of `MetaStudioToken` as an upgradable contracts; however all the reported issues are managed properly and therefore we classified them as false positives.

Code Documentation

1. **Fixed:** Consider adding an explanation on `MetaStudioToken.sol:_authorizeUpgrade` (L134-142) that this is mandatory for the `UUPSUpgradeable.sol` inheritance dependency.
2. **Fixed:** The NatSpec for `MetaStudioToken.sol:pause()` (L112) states that "No action available on the contract except `unpause`". The statement is not true. The `pause` function only stops transfer-related actions. Please update the documentation.

Adherence to Best Practices

1. **Fixed:** Check if the team can benefit from adding an index to the `TrustedForwarderChanged` event in the contract `ERC2771ContextUpgradeable.sol:L18`.
2. **Fixed:** Consider removing unnecessary functions overriding in `MetaStudioToken.sol` contract: `approve`, `allowance`, `balanceOf`, `decimals`, `name`, `symbol`, `totalSupply`, `transfer`, and `transferFrom`.
3. **Fixed:** The test coverage for the contract `ERC2771ContextUpgradeable.sol` is relatively low, with the branch coverage of 50%. We recommend adding tests to ensure the coverage is larger than 90%.
4. **Fixed:** In the documentation it is stated that no trusted forwarded is setup at deployment time. So there is no need to call `_setTrustedForwarder()` in `__ERC2771_init()`.

Test Results

Test Suite Results

All tests passed with running yarn coverage command.

```
MetaStudioToken
  ======== MetaStudioToken: Creation test ================================================
     ✓ Initial owner is mandatory
  ======== Contract: ERC165 ==============================================
    should support interface ERC165
      ERC165 is checking ERC165...
         ✓ interface is reported as supported
         ✓ interface's functions are in ABI
    ERC20 and extensions
      ERC165 is checking ERC20...
         ✓ interface is reported as supported (89ms)
         ✓ interface's functions are in ABI
      ERC165 is checking AccessControl...
         ✓ interface is reported as supported
         ✓ interface's functions are in ABI
      ERC165 is checking AccessControlEnumerable...
         ✓ interface is reported as supported
         ✓ interface's functions are in ABI
      ERC165 is checking Pausable...
         ✓ interface is reported as supported
         ✓ interface's functions are in ABI
    Others
      ERC165 is checking ERC2771...
         ✓ interface is reported as supported
         ✓ interface's functions are in ABI
      ERC165 is checking ERC1363...
         ✓ interface is reported as supported
         ✓ interface's functions are in ABI
  ======== Contract: ERC20 ===============================================
     ✓ has the good name
     ✓ has the good symbol
     ✓ has 18 decimals
    total supply
       ✓ returns the total amount of tokens
    balanceOf
      when the requested account has no tokens
         ✓ returns zero
      when the requested account has some tokens
         ✓ returns the total amount of tokens
    transfer
      when the recipient is not the zero address
        when the sender does not have enough balance
           ✓ reverts
        when the sender transfers all balance
           ✓ transfers the requested amount
```

```
        ✓ emits a transfer event
      when the sender transfers zero tokens
        ✓ transfers the requested amount
        ✓ emits a transfer event
    when the recipient is the zero address
      ✓ reverts
  transfer from
    when the token owner is not the zero address
      when the recipient is not the zero address
        when the spender has enough allowance
          when the token owner has enough balance
            ✓ transfers the requested amount
            ✓ decreases the spender allowance
            ✓ emits a transfer event
            ✓ emits an approval event
          when the token owner does not have enough balance
            ✓ reverts
        when the spender does not have enough allowance
          when the token owner has enough balance
            ✓ reverts
          when the token owner does not have enough balance
            ✓ reverts
        when the spender has unlimited allowance
          ✓ does not decrease the spender allowance
          ✓ does not emit an approval event
      when the recipient is the zero address
        ✓ reverts
    when the token owner is the zero address
      ✓ reverts
  approve
    when the spender is not the zero address
      when the sender has enough balance
        ✓ emits an approval event
        when there was no approved amount before
          ✓ approves the requested amount
        when the spender had an approved amount
          ✓ approves the requested amount and replaces the previous one
      when the sender does not have enough balance
        ✓ emits an approval event
        when there was no approved amount before
          ✓ approves the requested amount
        when the spender had an approved amount
          ✓ approves the requested amount and replaces the previous one
    when the spender is the zero address
      ✓ reverts
  decrease allowance
    when the spender is not the zero address
      when the sender has enough balance
        when there was no approved amount before
          ✓ reverts
        when the spender had an approved amount
          ✓ emits an approval event
          ✓ decreases the spender allowance subtracting the requested amount
          ✓ sets the allowance to zero when all allowance is removed
          ✓ reverts when more than the full allowance is removed
      when the sender does not have enough balance
        when there was no approved amount before
          ✓ reverts
        when the spender had an approved amount
          ✓ emits an approval event
          ✓ decreases the spender allowance subtracting the requested amount
          ✓ sets the allowance to zero when all allowance is removed
          ✓ reverts when more than the full allowance is removed
    when the spender is the zero address
      ✓ reverts
  increase allowance
    when the spender is not the zero address
      when the sender has enough balance
        ✓ emits an approval event
        when there was no approved amount before
          ✓ approves the requested amount
        when the spender had an approved amount
          ✓ increases the spender allowance adding the requested amount
      when the sender does not have enough balance
        ✓ emits an approval event
        when there was no approved amount before
          ✓ approves the requested amount
        when the spender had an approved amount
          ✓ increases the spender allowance adding the requested amount
    when the spender is the zero address
      ✓ reverts
======== Contract: AccessControl ================================================
  ERC165 is checking AccessControl...
    ✓ interface is reported as supported
    ✓ interface's functions are in ABI
  default admin
    ✓ deployer has default admin role
    ✓ other roles's admin is the default admin role
    ✓ default admin role's admin is itself
  granting
    ✓ non-admin cannot grant role to other accounts
    ✓ granting a role raise event RoleGranted
    ✓ accounts can be granted a role multiple times, but only one event
  revoking
    ✓ roles that are not had can be revoked
    with granted role
      ✓ admin can revoke role
      ✓ non-admin cannot revoke role
      ✓ a role can be revoked multiple times
  renouncing
    ✓ roles that are not had can be renounced
    with granted role
      ✓ bearer can renounce role
      ✓ only the sender can renounce their roles
      ✓ a role can be renounced multiple times
  onlyRole modifier
    ✓ do not revert if sender has role
    ✓ revert if sender doesn't have role #1
    ✓ revert if sender doesn't have role #2
  ERC165 is checking AccessControlEnumerable...
    ✓ interface is reported as supported
    ✓ interface's functions are in ABI
  enumerating
    ✓ role enumeration should be in sync after renounceRole call
======== Contract: Pausable =====================================================
  transfer
    ✓ allows to transfer when unpaused
    ✓ allows to transfer when paused and then unpaused
    ✓ reverts when trying to transfer when paused
  transfer from
    ✓ allows to transfer from when unpaused
    ✓ allows to transfer when paused and then unpaused
    ✓ reverts when trying to transfer from when paused
======== Contract: ERC1363 ======================================================
  via transferFromAndCall
    with data
      to a valid receiver contract
        ✓ should call onTransferReceived
    without data
      to a valid receiver contract
        ✓ should call onTransferReceived
    testing ERC20 behaviours
      when the sender does not have enough balance
        with data
          ✓ reverts
        without data
          ✓ reverts
      when the sender has enough balance
        with data
          ✓ transfers the requested amount
          ✓ emits a transfer event
        without data
          ✓ transfers the requested amount
```

```
            ✓ emits a transfer event
      to a receiver that is not a contract
         ✓ reverts
      to a receiver contract returning unexpected value
         ✓ reverts
      to a receiver contract that throws
         ✓ reverts
      to a contract that does not implement the required function
         ✓ reverts
    via transferAndCall
      with data
        to a valid receiver contract
           ✓ should call onTransferReceived
      without data
        to a valid receiver contract
           ✓ should call onTransferReceived
      testing ERC20 behaviours
        when the sender does not have enough balance
          with data
             ✓ reverts
          without data
             ✓ reverts
        when the sender has enough balance
          with data
             ✓ transfers the requested amount
             ✓ emits a transfer event
          without data
             ✓ transfers the requested amount
             ✓ emits a transfer event
      to a receiver that is not a contract
         ✓ reverts
      to a receiver contract returning unexpected value
         ✓ reverts
      to a receiver contract that throws
         ✓ reverts
      to a contract that does not implement the required function
         ✓ reverts
    via approveAndCall
      with data
        to a valid receiver contract
           ✓ should call onApprovalReceived
      without data
        to a valid receiver contract
           ✓ should call onApprovalReceived
      testing ERC20 behaviours
        with data
           ✓ approves the requested amount
           ✓ emits an approval event
        without data
           ✓ approves the requested amount
           ✓ emits an approval event
      to a spender that is not a contract
         ✓ reverts
      to a spender contract returning unexpected value
         ✓ reverts
      to a spender contract that throws
         ✓ reverts
      to a contract that does not implement the required function
         ✓ reverts
  ======== ERC20 VOTES ================================================
     ✓ initial nonce is 0
     ✓ domain separator
     ✓ minting restriction
    set delegation
      call
         ✓ delegation with balance (40ms)
         ✓ delegation without balance
      with signature
         ✓ accept signed delegation
         ✓ rejects reused signature
         ✓ rejects bad delegatee
         ✓ rejects bad nonce
         ✓ rejects expired permit
    change delegation
       ✓ call (51ms)
    transfers
       ✓ no delegation
       ✓ sender delegation
       ✓ receiver delegation
       ✓ full delegation
    Compound test suite
      balanceOf
         ✓ grants to initial account
      numCheckpoints
         ✓ returns the number of checkpoints for a delegate (107ms)
         ✓ does not add more than one checkpoint in a block (88ms)
      getPastVotes
         ✓ reverts if block number >= current block
         ✓ returns 0 if there are no checkpoints
         ✓ returns the latest block if >= last checkpoint block
         ✓ returns zero if < first checkpoint block
         ✓ generally returns the voting balance at the appropriate checkpoint (98ms)
    getPastTotalSupply
       ✓ reverts if block number >= current block
       ✓ returns 0 if there are no checkpoints
       ✓ returns the latest block if >= last checkpoint block
       ✓ returns zero if < first checkpoint block
  ======== Contract: ERC20 Permit ================================================
     ✓ initial nonce is 0
     ✓ domain separator
    when msg signer == owner
       ✓ ERC20 Approval event is emitted
       ✓ nonce for owner should be incremented by 1
       ✓ Allowance(owner, spender) should be equal to value
       ✓ rejects reused signature
       ✓ rejects expired permit
    when msg signer != owner
       ✓ rejects other signature
       ✓ reverts if owner is zeroAddress
       ✓ reverts if spender is not the approved spender
  ======== MetaStudioToken: Specific tests ================================================
    Chain Id
       ✓ get
    AccessControl
      access control for Pause
        role is not granted
           ✓ pause is disallowed
           ✓ unpause is disallowed
        role is granted
           ✓ pause is allowed
           ✓ unpause is allowed
      access control for Forwarder
        role IS NOT granted
           ✓ setting a trusted forwarder is not allowed"
        role IS granted
           ✓ setting a trusted forwarder is allowed"
  ======== ERC2771 ================================================
    no trusted forwarder defined
       ✓ unrecognize trusted forwarder
       ✓ setting a trusted forwarder should emit "TrustedForwarderChanged"
    a trusted forwarder is defined
       ✓ recognize trusted forwarder
      forwarding ERC20
        transfer
          when the recipient is not the zero address
            when the sender does not have enough balance
               ✓ reverts
            when the sender transfers all balance
               ✓ transfers the requested amount
               ✓ emits a transfer event
            when the sender transfers zero tokens
               ✓ transfers the requested amount
               ✓ emits a transfer event
```

```
    when the recipient is the zero address
      ✓ reverts
  transfer from
    when the token owner is not the zero address
      when the recipient is not the zero address
        when the spender has enough allowance
          when the token owner has enough balance
            ✓ transfers the requested amount
            ✓ decreases the spender allowance
            ✓ emits a transfer event
            ✓ emits an approval event
          when the token owner does not have enough balance
            ✓ reverts
        when the spender does not have enough allowance
          when the token owner has enough balance
            ✓ reverts
          when the token owner does not have enough balance
            ✓ reverts
        when the spender has unlimited allowance
          ✓ does not decrease the spender allowance
          ✓ does not emit an approval event
      when the recipient is the zero address
        ✓ reverts
    when the token owner is the zero address
      ✓ reverts
  approve
    when the spender is not the zero address
      when the sender has enough balance
        ✓ emits an approval event
        when there was no approved amount before
          ✓ approves the requested amount
        when the spender had an approved amount
          ✓ approves the requested amount and replaces the previous one
      when the sender does not have enough balance
        ✓ emits an approval event
        when there was no approved amount before
          ✓ approves the requested amount
        when the spender had an approved amount
          ✓ approves the requested amount and replaces the previous one
    when the spender is the zero address
      ✓ reverts


  196 passing (6s)
```

Code Coverage

The code coverage is very good for all contracts except for the `ERC2771ContextUpgradeable` which decreases the overall coverage.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
ERC1363/	100	91.67	100	100	
ERC1363Upgradeable.sol	100	91.67	100	100	
Proxy/	100	100	100	100	
Import.sol	100	100	100	100	
metatx/	72.73	50	83.33	75	
ERC2771ContextUpgradeable.sol	72.73	50	83.33	75	65,66,68
IERC2771Upgradeable.sol	100	100	100	100	
mocks/	95.45	87.5	100	100	
ERC1363ReceiverMock.sol	100	100	100	100	
ERC1363SpenderMock.sol	100	100	100	100	
ERC2771ForwarderMock.sol	91.67	75	100	100	
tokens/	92.59	50	78.57	92.59	
IPausable.sol	100	100	100	100	
MetaStudioToken.sol	92.59	50	78.57	92.59	161,194
All files	**92.86**	**80.77**	**89.47**	**94.12**	

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.

Contracts

```
2208bdbce7ce03ab7fbe60cd6dc77b46cef9a6ab22aca7743c5e49b130c9115d ./contracts/tokens/MetaStudioToken.sol
d32d1c25de875495effe7923678f70f7c1eed23ba62883ed4327edcf0903f21f ./contracts/tokens/IPausable.sol
7f57a2092524126b91a0a082759f190e9593073c88d5e7eab98fa0d805cb7522 ./contracts/metatx/IERC2771Upgradeable.sol
86715e44863fe6fb1d74efdbf2c873a0cb92c78e9be9793c876535d62c49e6ec ./contracts/metatx/ERC2771ContextUpgradeable.sol
afcd362926639e0f607ee88b73e8ae2e87743fbaebfbd6622d86ccc9a86e5a33 ./contracts/ERC1363/ERC1363Upgradeable.sol
66e3f95b3b9a17c1e636a88171df6f16efe82e972df905be7e69e523a3529b8c ./contracts/Proxy/Import.sol
265ed46749b632fbe8958a7b21a7b94f7c9917420d20cf636af08b6280322f3c ./contracts/mocks/ERC1363SpenderMock.sol
```

Tests

```
fba39b4ca1332e4e9f208ebedee7509aca1d3602e8ae2881088a0d960e03a143 ./test/chai-setup.ts
d9b69afc64555dc4b2321d475f509e26275b4296573a7274991105cea2c7a815 ./test/helpers/eip712.ts
bb99a9a678652fe598335ed5a19b6790883b7c97bc0c569983ff3f49df9f94bf ./test/tokens/MetaStudioToken.test.ts
2e2bee25f28f8a6d4c83df31376ecbe30225b9d433cec4cdb38da616e7de2837 ./test/tokens/tests/ERC20.test.ts
7095a85114dcaba254bb913f41400ae82250045e454c5196e1f00bdbab191fe3 ./test/tokens/tests/ERC165.test.ts
ee3fd17b9f4cea03dbff8ec5e516f53564f113e5c0ac97885fb3d041d41ca016 ./test/tokens/tests/ERC20Votes.test.ts
58d18e2ec7ec5554fbd982b29d7c1b05072338b6ae1c7a4a37948d3baf8393b7 ./test/tokens/tests/AccessControl.test.ts
a6a3694d76f184792750573210d99b44ded61db70edb78c9fba1b73e2fb3fd6f ./test/tokens/tests/ERC20Permit.test.ts
44688b4652193b84dff881c880d89b8ef11425f6531a1c81c7daff27e9fe3945 ./test/tokens/tests/Pausable.test.ts
d809f400b225f9cdd06171569d389efa5ee9aac2a3b637f3c013e764e8b247e1 ./test/tokens/tests/ERC2771.test.ts
e2f5580215e72c673e242333d1504ab2e0a6ae89bb1eac542c54d3b44c1eacd7 ./test/tokens/tests/ERC1363.test.ts
091daedf0ec6e53bb92e9ab6f3b8859baf1c43f2cb0f9bb9daa9f60270e8754a ./test/tokens/tests/behaviors/ERC2771-ERC20.behavior.ts
aec6a67d9cac252e18077e176117ab725fe9e0506550571269a55eb4c822ff46 ./test/tokens/tests/behaviors/SupportsInterface.behavior.ts
85a742164fdc82de76e0a16d685255d1fa568e26e52c612ced1e1e530766134c ./test/tokens/tests/behaviors/ERC20.behavior.ts
eed332526c330763816dfef08d97fcf31af2606a802e247418e9c19f6b2eb993 ./test/tokens/tests/behaviors/AccessControl.behavior.ts
c8ed66a0e1b9e7cf1a7599d3ae00b88d7ce653118457d727f8a0c0e72e8f0145 ./test/shared/contexts.ts
7bc6328082596839826989a2e96b310afb791b2613846f2903892351fd3f4439 ./test/shared/utils.ts
18399af61fc35f312d05b35e10727a8a50a1dab72511ba01a85ea3fa8407186a ./test/shared/types.ts
11d992d8027e8eb78f10d4437bee98d4ccabd08d1a0a5ca87c6f78a1a859705e ./test/shared/constants.ts
bb71278bda87532e8ba037fac3a1f246762bcbc927a405f1d2258b2ce8d8ff7e ./test/utils/index.ts
```

Changelog

- 2022-07-22 - Initial report
- 2022-08-05 - Revised report based on commit `8d130d4`

About Quantstamp

Quantstamp is a Y Combinator-backed company that helps to secure blockchain platforms at scale using computer-aided reasoning tools, with a mission to help boost the adoption of this exponentially growing technology.

With over 1000 Google scholar citations and numerous published papers, Quantstamp's team has decades of combined experience in formal verification, static analysis, and software verification. Quantstamp has also developed a protocol to help smart contract developers and projects worldwide to perform cost-effective smart contract security scans.

To date, Quantstamp has protected $5B in digital asset risk from hackers and assisted dozens of blockchain projects globally through its white glove security assessment services. As an evangelist of the blockchain ecosystem, Quantstamp assists core infrastructure projects and leading community initiatives such as the Ethereum Community Fund to expedite the adoption of blockchain technology.

Quantstamp's collaborations with leading academic institutions such as the National University of Singapore and MIT (Massachusetts Institute of Technology) reflect our commitment to research, development, and enabling world-class blockchain security.