Quantstamp Security Assessment Certificate

July 12th 2021 — Quantstamp Verified

# Coinvise

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

## Executive Summary

| | |
|---|---|
| Type | Social Tokens and NFTs |
| Auditors | Jose Ignacio Orlicki, Senior Engineer<br>Joseph Xu, Technical R&D Advisor<br>Fayçal Lalidji, Security Auditor |
| Timeline | 2021-05-11 through 2021-06-20 |
| EVM | Berlin |
| Languages | Solidity |
| Methods | Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review |
| Specification | Coinvise Wiki |
| Documentation Quality | Medium |
| Test Quality | Medium |

Source Code

| Repository | Commit |
|---|---|
| coinvise-contracts | d0c58fc (audit) |
| coinvise-contracts | b5f5c1f (reaudit) |

| | |
|---|---|
| Total Issues | 21 (15 Resolved) |
| High Risk Issues | 1 (1 Resolved) |
| Medium Risk Issues | 7 (5 Resolved) |
| Low Risk Issues | 9 (7 Resolved) |
| Informational Risk Issues | 4 (2 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). |
| Resolved | 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

We have reviewed the code, documentation, and test suite and found several issues of various severities. Overall, we consider the code to be well-written but with insufficient documentation and a suboptimal testing suite. The documentation is very bare-bones with little technical detail. The Wiki gives an idea of how the app is supposed to work from the front-end aspect. The role of each contract should be further documented and readily available. Currently, the test suite is not in good shape, at least 1 test is failing and the code coverage can be dramatically improved. We have outlined suggestions to better follow best practices, and recommend addressing all the findings to tighten the contracts for future deployments or contract updates. We also provide suggestions for improvements to follow the best practices. We recommend addressing all the **21** findings to harden the contracts for future deployments or contract updates. We recommend against deploying the code as-is.

**After reaudit:** Quantstamp has performed a reaudit to check the proposed fixes. All of the previously identified issues were either resolved (15 issues) or acknowledged (6 issues). Tests are very exhaustive but *not all tests are* currently passing. During reaudit we found Hardhat Coverage module was not working. Newly added contracts were not audited, best practice is not to add new functionality on reaudits: `NFTProxy`, `TokenLinearBondedEthDeployer`, `Erc20TokenLinearBondedErc20Payable`, `Erc20TokenLinearBondedEthPayable`, `Erc20TokenSigmoidBondedErc20Payable`, `Erc20TokenSigmoidBondedEthPayable`.

| ID | Description | Severity | Status |
|---|---|---|---|
| QSP-1 | Truncation Error Exploit | High | Fixed |
| QSP-2 | Signer Address Exploit | Medium | Fixed |
| QSP-3 | `ERC721Token` Not Enforcing `onlyOwner` | Medium | Acknowledged |
| QSP-4 | Duplicated Names and Symbols Allowed | Medium | Acknowledged |
| QSP-5 | Reentrancy and Arbitrary Token Contracts | Medium | Fixed |
| QSP-6 | Integer Overflow of Financial Variables | Medium | Fixed |
| QSP-7 | Truncation Error in `Coinvise.depositToken` | Medium | Fixed |
| QSP-8 | Unwithdrawable Unsold Deposit | Medium | Fixed |
| QSP-9 | Denial-of-Service (DoS) | Low | Mitigated |
| QSP-10 | Dusting | Low | Acknowledged |
| QSP-11 | Price Checks Missing | Low | Fixed |
| QSP-12 | Privileged Roles and Ownership | Low | Mitigated |
| QSP-13 | Gas Consumption Concerns for Unbounded Structures | Low | Acknowledged |
| QSP-14 | ERC20 Might Be Unpausable | Low | Fixed |
| QSP-15 | Missing Input Validations | Low | Mitigated |
| QSP-16 | Fee Deposit | Low | Fixed |
| QSP-17 | Truncation Error in `NFTMarketPlace._calculateCut` | Low | Fixed |
| QSP-18 | Uncommon Pattern for Fees | Informational | Acknowledged |
| QSP-19 | Unlocked Pragma | Informational | Fixed |
| QSP-20 | Missing Gasless `ecrecover` Checks | Informational | Acknowledged |
| QSP-21 | Allowance Double-Spend Exploit | Informational | Mitigated |

# Quantstamp Audit Breakdown

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

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 v0.7.0
- Mythril v0.22.16

Steps taken to run the tools:

1. Installed the Slither tool: `pip install slither-analyzer`
2. Run Slither from the project directory: `slither .`
3. Installed the Mythril tool from Pypi: `pip3 install mythril`
4. Ran the Mythril tool on each contract: `myth -x path/to/contract`

# Findings

## QSP-1 Truncation Error Exploit

Severity: *High Risk*

**Status:** Fixed

**File(s) affected:** `Coinvise.sol`

**Description:** In `Coinvise.buyToken` L526 a truncation error can be exploited by attackers to withdraw users deposits without paying ETH. As it can be seen in the code, the token amount to be withdrawn is divided by the `decimalsZero`, which will truncate any value lower than `10**Token.decimals()` to zero, `totalPrice` value will be set to zero allowing the attacker to deposit zero ETH even if the price was defined correctly by the depositor. Depending on the deposit amount an attacker can make multiple transactions with an amount value lower than `decimalsZeros` to withdraw all users' funds without paying any ETH.

**Recommendation:** We recommend fixing the computation error in L526 by modifying the ordering of the arithmetical operations, multiplications should always be executed before divisions.

**Update:** Fixed in this commit. Project Note is *"Modified the ordering of the arithmetical operations. Added Require a price for a buyToken operation."*

## QSP-2 Signer Address Exploit

Severity: *Medium Risk*

**Status:** Fixed

**File(s) affected:** `NFTAirdrop.sol`, `Coinvise.sol`

**Description:** In `NFTAirdrop.claim` and `Coinvise.claim` following the `r`, `s` and `v` inputs, the `signer` address result can be manipulated to be the `0x0` address. If for any reason `trustedAddress` is set to zero by the owner or if the address is not correctly initialized, attackers may claim all NFTs available from any campaign.

**Recommendation:** Add a requirement that checks if the signer address is different the the `0x0` address.

**Update:** Fixed in this commit. Project Note is *"Added require to check that the recovered address is not 0x0. Added require so the trustedAddress is not set as 0x0."*

## QSP-3 `ERC721Token` Not Enforcing `onlyOwner`

Severity: *Medium Risk*

**Status:** Acknowledged

**File(s) affected:** `Erc721Token.sol`

**Description:** `ERC721Token` inherits from `Ownable` but to enforce ownership on specific functions, `onlyOwner` modifier must be added to the owner-critical functions such as `mintTo()` and `mint()`. We are assuming that the owner is minting a set of NFTs from the same class and only he should be allowed to mint extra NFTs of this class. Therefore, not using `ownerOnly` is allowing anyone to mint NFTs of the collection created by the owner.

**Recommendation:** Add `onlyOwner` modifier if applicable for all affected functions. We also recommend documenting who can mint NFTs and ERC20.

**Update:** Acknowledged by the team. Project Note is *"Anyone is allowed to mint NFTs for the collection ("Coinvise" collection). Added comment in the contract file."*

## QSP-4 Duplicated Names and Symbols Allowed

Severity: *Medium Risk*

**Status:** Acknowledged

**File(s) affected:** `NFTMarketplace.sol`, `TokenDeployer.sol`

**Description:** Validation of a listing in `NFTMarketplace.sol` may not be correct in `buyNftInEth()` and `buyNftInErc20Tokens()` because the same NFT can be listed multiple times for ETH or ERC20s under different IDs. Also, unlisting only clears the entry in `_listedNftTokenIds EnumerableSet.Uintset` and does not clear the `nftListingById` mapping. The same NFT can occupy multiple `nftListingById` mapping if it gets listed after being delisted. Meanwhile, contract `TokenDeployer` allows any users to deploy ERC20Token but there is no check for duplicated names or symbols. This will allow for a malicious attack to deploy duplicated tokens with the same name or symbol to mislead users of the platform.

**Exploit Scenario:**

1. A seller lists an NFT for 1 ETH and also for 4000 USDC. This is the same NFT but both listings result in a different `_nftListingId`.
2. A buyer buys the NFT for 4000 USDC (say 1 ETH is at 4200). The USDC `_nftListingId` is cleared from `EnumerableSet.UintSet _listedNftTokenIds`, but the ETH `_nftListingId` is still present. Both data are still in the `nftListingById` mapping as well.
3. ETH dumps to 3000 USD, which causes another buyer to want to buy the same NFT for 1 ETH. The ETH `_nftListingId` is validated instead of the USDC `_nftListingId`, so the NFT does not appear as already delisted.
4. Fortunately, the first buyer won't lose the NFT since the transfer would fail and revert (`nftListing.listedBy` address does not hold the NFT anymore). However, the new buyer will waste gas.

```
nftContract.safeTransferFrom(
  nftListing.listedBy,
  _sender,
  nftListing.nftTokenId
);
```

**Recommendation:** Manage the marketplace NFT listing by address instead of by ID. Otherwise, make sure to unlist the same NFT by checking the listings of the `nftListing.listedBy` address (although gas consumption might be a concern in this case). For `TokenDeployer`, similarly, consider including a mapping of existing names and/or symbols to fail in the case of duplicates names and/or symbols.

**Update:** Acknowledged by the team and some changes in this commit. Now function `_unlistNft()` removes the listing ID from `nftListingById` mapping. However, duplicate names and symbols are possible at the contract level. Project Note is

> *By design, a listing has a currency (token) associated, enabling multiple listings for the same NFTs, controlling those settings in our software. We accept that anyone interacting directly with the SC trying to trick the system wastes some gas. Removing the element from nftListingById mapping. TokenDeployer intention is to provide erc20 token deployment capabilities on MetaTx environments, not to add any other requirement/check that the underlying network.*

## QSP-5 Reentrancy and Arbitrary Token Contracts

Severity: *Medium Risk*

**Status:** Fixed

**File(s) affected:** `NFTAirdrop.sol`, `NFTMarketplace.sol`, `Distribution.sol`, `Marketplace.sol`

**Description:** `NFTAirdrop.createCampaign()` is `external view` and has no restriction on the input contract `_tokenAddress` or `msg.sender`. This makes it an easy target for reentrancy attacks. Arbitrary malicious tokens implementing the ERC721 interface will try to call other functions of the platform to duplicate transactions or manipulate balances. Similar with `NFTMarketplace.listNftInEth()`, having arbitrary `_nftTokenAddress` parameter. A similar attack is feasible for `Distribution.sol` and `Marketplace.sol` but suited for malicious external ERC20 tokens.

**Recommendation:** To solve or mitigate these issues your can either whitelist the addresses for `_tokenAddress/_nftTokenAddress` or you have to harden the contract for reentrancy attacks with a lock on every critical function (such as ReentrancyGuard, for example). If there are no plans for this function to be called externally from other contracts, then document this and warn against this type of usage.

**Update:** Fixed in this commit. Project Note is *"Added ReentrancyGuard in critical functions."*

## QSP-6 Integer Overflow of Financial Variables

### Severity: *Medium Risk*

**Status:** Fixed

**File(s) affected:** `EthPool.sol`, `NFTAirdrop.sol`

**Description:** Integer overflow/underflow occurs when an integer hits its bit-size limit. Every integer has a set range; when that range is passed, the value loops back around. A clock is a good analogy: at 11:59, the minute hand goes to 0, not 60, because 59 is the largest possible minute. Integer overflow and underflow may cause many unexpected kinds of behavior and was the core reason for the `batchOverflow` attack. Here's an example with `uint8` variables, meaning unsigned integers with a range of `0..255`.

```
        function under_over_flow() public {
            uint8 num_players = 0;
            num_players = num_players - 1;  // 0 - 1 now equals 255!
            if (num_players == 255) {
                emit LogUnderflow();        // underflow occurred
            }
            uint8 jackpot = 255;
            jackpot = jackpot + 1;          // 255 + 1 now equals 0!
            if (jackpot == 0) {
                emit LogOverflow();         // overflow occurred
            }
        }
```

The affected function is `receiveMaticEquivalent()` that is `payable` and the calculation of `expectedEthWeiAmount` can overflow. Same for `NFTAirdrop.createCampaign()` with parameter `_linksAmount` that is vulnerable and can overflow financial variable `priceInWei`.

**Recommendation:** Use `SafeMath` or any popular variant, to have secure arithmetical operations that fail on overflow and underflow. Consider using this type of secure arithmetic anywhere there is a complex calculation.

**Update:** Fixed in this commit. Project Note is *"Using SafeMath in arithmetic operations."*

## QSP-7 Truncation Error in `Coinvise.depositToken`

### Severity: *Medium Risk*

**Status:** Fixed

**File(s) affected:** `Coinvise.sol`

**Description:** The formula implemented in `Coinvise.depositToken` to calculate the user's deposit fees truncates `_amount` when dividing it by `decimalsZeros`, this error will lead users to pay less or no fees. The formula development should be clearly documented since the current state does not allow one to assess its correctness.

**Recommendation:** Reimplement the formula while adding enough inline documentation to assess its correctness.

**Update:** Fixed in this commit. Project Note is *"Reformulation, avoiding truncation. Added comment with formulaAdded require to make sure the price to pay > 0."*

## QSP-8 Unwithdrawable Unsold Deposit

### Severity: *Medium Risk*

**Status:** Fixed

**File(s) affected:** `Coinvise.sol`

**Description:** In `Coinvise` contract all unsold deposits cannot be withdrawn. There is no function in the contract to allow owners of deposits to withdraw their funds. Only the amount sold will be withdrawable through `Coinvise.withdrawDepositOwnerBalance`.

**Recommendation:** Either document this behavior or implement a function allowing depositors to withdraw their unsold funds.

**Update:** Fixed in this commit. Project Note is *"Added a `withdrawDeposit()` function."*

## QSP-9 Denial-of-Service (DoS)

### Severity: *Low Risk*

**Status:** Mitigated

**File(s) affected:** `NFTMarketplace.sol`

**Description:** A Denial-of-Service (DoS) attack is a situation in which an attacker renders a smart contract unusable. Contract `NFTMarketplace` allows any user to list more NFTs, with an identifier recorded on L222 as `_listedNftTokenIds.add(nftListing.listingId);`. For usability, you have to view the state of the current market with `getCurrentNftListingIds()`. The later function eventually will revert if a malicious user or any user adds enough NFTs to consume the gas limit of transactions. Also, maintaining the contract state off-chain is very difficult because the events emitted when listing an NFT have very little information. Such an event is `emit NftListed(nftListing.listingId);`, including only the identification and not other information.

**Recommendation:** Using mappings to maintain the state of NFTs (and everything) in the contract and emit more detailed Solidity Events so off-chain databases can have an accurate view of the state of the contract.

**Update:** Mitigated in this commit. Possible to address using off-chain DB with the emitted events. `NFTMarketplace.getCurrentNftListingIds()` may still revert if spammed. Project Note is *"Added details on NftListed event, so blockchain info can be replicated off-chain, even though we're currently maintaining the state of the marketoff-chain without the need of this, as all actions take place in our platform. Currently, we are not using getCurrentNftListingIds()."*

## QSP-10 Dusting

### Severity: *Low Risk*

**Status:** Acknowledged

**File(s) affected:** `Coinvise.sol`

**Description:** Possible dusting on `Coinvise.sol` if `_amountPerLink` is not specified properly by the campaign manager in `createCampaign()`.

**Recommendation:** Include basic parameter check to impose minimum `_amountPerLink`.

**Update:** Acknowledged by the team. Project Note is *"Accepting by design."*.

## QSP-11 Price Checks Missing

Severity: *Low Risk*

**Status:** Fixed

**File(s) affected:** `NFTMarketplace.sol`

**Description:** Lack of minimal price checks can lead to a seller incorrectly specifying a very low or very high listing price on `NFTMarketplace.sol`. This may be especially problematic if the NFT gets listed with ERC20 tokens whose decimals are not standardized (i.e., non-18 decimals). Notably, NFTs can be listed for a price of zero.

**Recommendation:** Add minimum price validity checks, or document how is this checked in the front-end if applicable.

**Update:** Fixed in this commit. Project Note is *"Requires min price > 0. We allow anything else by design."*

## QSP-12 Privileged Roles and Ownership

Severity: *Low Risk*

**Status:** Mitigated

**File(s) affected:** `Coinvise.sol`, `NFTMarketplace.sol`, `EthPool.sol`

**Description:** Smart contracts will often have `owner` variables to designate the person with special privileges to make modifications to the smart contract.

- The owner can change premiums on `Coinvise.sol` and premium percentages on `NFTMarketplace.sol` at any time with little restrictions on the parameter values.
- The owner can pause the marketplace, as well as whitelist/blacklist payment tokens.
- On `EthPool`, the owner of the contract can withdraw ETH funds to arbitrary addresses.

**Recommendation:** This centralization of power needs to be made clear to the users, especially depending on the level of privilege the contract allows to the owner. Also, include basic parameter checks to prohibit excessive premiums. Currently, the fee structure seems to be 5% on deposit (SAFE) and 0.2% per Airdrop, so set a maximum premium to a reasonable multiple of those. For other privileges, document these clearly to users.

**Update:** Mitigated in this commit. Project Note is *"Set max. deposit fee percentage."*

## QSP-13 Gas Consumption Concerns for Unbounded Structures

Severity: *Low Risk*

**Status:** Acknowledged

**File(s) affected:** `NFTAirdrop.sol`, `Distribution.sol`

**Description:** Gas usage is an important concern for smart contract developers and users, since high gas costs may prevent users from wanting to use the smart contract. Even worse, some gas usage issues may prevent the contract from providing services entirely. For example, if an unbounded structure, such as an array, requires too much gas to exit, then it may prevent the contract from functioning correctly entirely. Function `getCampaignIdsFromManager()` is affected because it involves copying the array `campaignIds[_campaignManager]` from EVM storage to EVM memory. Same for `getCampaign()` that involves copying the array `_campaign.tokenIds` from EVM storage to EVM memory.

**Recommendation:** It is best to break such structures into individual chunks as possible, or explicitly limit the size of the structures trying to avoid denial-of-service attacks also.

**Update:** Acknowledged by the team. Project Note is *"All those methods are not used. might be better deleting them."*

## QSP-14 ERC20 Might Be Unpausable

Severity: *Low Risk*

**Status:** Fixed

**File(s) affected:** `ERC20Token.sol`

**Description:** This is a particular case of missing input validations but due to its impact is reported separately. In case there is no explicit `_minter != address(0)` in `ERC20Token` (`_mint == address(0)`), then `pause()` and `unpause()` will fail.

**Recommendation:** Consider in the constructor to set the `_minter` as `msg.sender` if not explicit minter is set like `_minter = minter != address(0) ? minter : msg.sender;`.

**Update:** Fixed in this commit. Project Note is *"Setting minter as msg.sender"*

## QSP-15 Missing Input Validations

Severity: *Low Risk*

**Status:** Mitigated

**File(s) affected:** `ERC20Token.sol`, `Erc721Token.sol`, `NFTAirdrop.sol`, `Coinvise.sol`, `NFTMarketPlace.sol`

**Description:** Checking inputs in smart contracts will allow the avoidance of invalid internal states or spending gas in generating invalid instances that are not useful. It is important to perform input validation on the setter functions too (such as `trustedAddress != address(0)`). Some cases:

- In `ERC20Token.constructor()` verify that `supplyCap > 0` or `supplyCap > SOME_MINIMUM`. Same for `value` in `ERC20Token._mint()`
- In `ERC721Token.mintTo()` verify that `_tokenId != ''` and that `_tokenId != address(0)`. Also check `_tokenId != ''` for `mint()`.
- `NFTAirdrop.initialize()` and setters needs more checks.
- `Coinvise.initialize()` and setters needs more checks.
- `NFTMarketPlace.initialize()` and setters needs more checks.

**Recommendation:** Include all possible validations for all inputs, unless this hurts the gas consumptions or contract size.

**Update:** Mitigated in this commit. Recommended cases are either acknowledged or unresolved, but added extra checks in other parts of the code. Project Note is

> `ERC20Token.constructor()` *is used by another contract with no initial supply. Added validations when needed. Allowing 0 values on fees, as it might be interesting business-wise.*

## QSP-16 Fee Deposit

Severity: *Low Risk*

**Status:** Fixed

**File(s) affected:** `Coinvise.sol`

**Description:** `Coinvise.createCampaignMeta` allows creation of campaigns without ETH deposit as opposed to `Coinvise.createCampaign`. The same issue is applicable for `multisend` and `multisendMeta`.

**Recommendation:** This behavior should be clearly documented since it allows user to execute transactions without paying any fee.

**Update:** Fixed in this commit. Unused `createCampaignMeta()` and `multisendMeta ()` methods removed. Project Note is *"Commented out methods"*.

## QSP-17 Truncation Error in `NFTMarketPlace._calculateCut`

Severity: *Low Risk*

**Status:** Fixed

**File(s) affected:** `NFTMarketPlace.sol`

**Description:** The formula implemented in `NFTMarketPlace._calculateCut` to calculate the fees truncates `_amount` when dividing it by `(10** (premiumPercentageDecimals.add(2)))`.

**Recommendation:** Multiplication operations should be executed first before any division.

**Update:** Fixed in this commit. Project Note is *"Modified order"*.

## QSP-18 Uncommon Pattern for Fees

Severity: *Informational*

**Status:** Acknowledged

**Description:** The contracts specify < 1% fee in a non-standard way. Specifically, `Coinvise.sol` and `NFTMarketplace.sol` use a combination of two variables `premiumPercentage` and `premiumPercentageDecimals` (in the case of `NFTMarketplace.sol`). A more typical approach is to use basis points (1 bp = 0.01%) for fee calculation.

**Recommendation:** For example, consider making this more standard, resulting in Coinvise::L460-465 being something like:

```
uint256 priceInWei =
  _price
    .mul(_amount.div(decimalsZeros))
    .mul(feeBasisPoint)
    .div(10000);
```

The variable above is also best re-named `uint256 depositFeeInWei`.

**Update:** Acknowledged by the team.

## QSP-19 Unlocked Pragma

Severity: *Informational*

**Status:** Fixed

**File(s) affected:** `EthPool.sol`, `NFTAirdrop.sol`, `TokenDeployer.sol`, `MaticToken.sol`, `etc.`

**Description:** Every Solidity file specifies in the header a version number of the format `pragma solidity (^)0.7.4`. 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, it is recommended to remove the caret to lock the file onto a specific Solidity version. Hardhat locks the project to a specific Solidity version but this issue still applies as is very common for projects to re-use contracts from other projects.

**Update:** Fixed in this commit. Version locked in 0.7.4. Project Note was *"Locked"*.

## QSP-20 Missing Gasless `ecrecover` Checks

Severity: *Informational*

**Status:** Acknowledged

**File(s) affected:** `EIP712MetaTransaction.sol`, `NFTAirdrop.sol`

**Description:** The function `executeMetaTransaction()` calls `verify()` on `EIP712MetaTransaction` that invokes `ecrecover` without safety checks on `_v`, `_r`, and `_s`, which may allow for signature malleability. Same for `claim()` in `NFTAirdrop`.

**Recommendation:** Use off-the-shelf libraries such as OpenZeppelin's ECDSA.sol, or clone the functionality with criteria and documenting this decision.

**Update:** Flagged by the client as *Unresolved* but changed to *Acknowledged* by Quantstamp team as malleability will not affect functions tracking transactions with nonces.

## QSP-21 Allowance Double-Spend Exploit

Severity: *Informational*

**Status:** Mitigated

**File(s) affected:** `Erc20Token.sol`

**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()`.
Pending community agreement on an ERC standard that would protect against this exploit, 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. Teams who decide to wait for such a standard should make these recommendations to app developers who work with their token contract.

**Update:** Already mitigated by recommended allowance functions.

## Automated Analyses

### Slither

Slither has detected many results out of which the majority have been filtered out as false positives and the rest have been integrated into the findings from this report.

### Mythril

Mythril has detected many results out of which the majority have been filtered out as false positives and the rest have been integrated into the findings from this report.

## Adherence to Best Practices

1. `mapping(uint256 => mapping(string => bool)) internal linksRewarded;` is unused in `Coinvise.sol` and `NFTAirdrop.sol`. *Unresolved*
2. Magic number usage in `_calculateCut()` in `NFTMarketplace.sol`. Parameterize and avoid magic numbers. This is also relevant to the non-standard implementation of fees issue. *Unresolved*
3. `Coinvise.getDepositId()` doesn't just get the deposit ID but also increments the next deposit ID. Either rename the function to something like `getAndIncrementDepositId()` or create separate functions for getting and incrementing the deposit ID. This comment applies to other functions that serve both get and increment functionalities on IDs. *Unresolved*
4. In `NFTMarketplace.sol`, there is an outstanding //TODO item with an unclear intention on L42. *Unresolved*
5. Minor changes from metatx-standard to make it *Unresolved*upgradeable using the OpenZeppelin contracts for ERC721. Document any changes on popular contracts.
6. Remove commented code and events in `contracts/TokenDeployer.sol`. *Fixed*

## Test Results

### Test Suite Results

`coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e jose$ npm run test`

```
coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e jose$ npm run test

> coinvise-contracts@1.0.0 test
> npx hardhat test

Compiling 9 files with 0.7.4
contracts/flatten/Erc20Token.sol:381:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor (string memory name_, string memory symbol_) public {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/Erc20Token.sol:708:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor () internal {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/Erc20TokenLinearBondedEthPayable.sol:2407:1: Warning: This contract has a payable fallback function, but no receive ether function. Consider adding a receive ether function.
contract Erc20TokenLinearBoundedEthPayable is ERC20Token {
^ (Relevant source part starts here and spans across multiple lines).
contracts/flatten/Erc20TokenLinearBondedEthPayable.sol:2490:3: The payable fallback function is defined here.
  fallback() external payable {
  ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/Erc20TokenLinearBondedEthPayable.sol:381:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor (string memory name_, string memory symbol_) public {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/Erc20TokenLinearBondedEthPayable.sol:708:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor () internal {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/Erc20TokenLinearBondedEthPayable.sol:2465:5: Warning: Unused local variable.
    uint256 m = slope;
    ^-------^

contracts/flatten/Erc721Token.sol:292:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor () internal {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/Erc721Token.sol:1431:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor (string memory name_, string memory symbol_) public {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/Erc721Token.sol:1844:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor () internal {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/Erc721Token.sol:1910:3: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
  constructor(string memory name, string memory version) public {
  ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/NFTAirdrop.sol:2983:5: Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
    address operator,
    ^--------------^
```

```
contracts/flatten/NFTAirdrop.sol:2984:5: Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
    address from,
    ^----------^

contracts/flatten/NFTAirdrop.sol:2985:5: Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
    uint256 tokenId,
    ^-------------^

contracts/flatten/NFTAirdrop.sol:2986:5: Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
    bytes calldata data
    ^-----------------^

contracts/flatten/TokenDeployer.sol:25:3: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
  constructor(string memory name, string memory version) public {
  ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenDeployer.sol:583:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor (string memory name_, string memory symbol_) public {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenDeployer.sol:910:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor () internal {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenDeployer.sol:1093:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor () internal {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenLinearBondedEthDeployer.sol:2609:1: Warning: This contract has a payable fallback function, but no receive ether function. Consider adding a receive ether function.
contract Erc20TokenLinearBoundedEthPayable is ERC20Token {
^ (Relevant source part starts here and spans across multiple lines).
contracts/flatten/TokenLinearBondedEthDeployer.sol:2692:3: The payable fallback function is defined here.
  fallback() external payable {
  ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenLinearBondedEthDeployer.sol:25:3: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
  constructor(string memory name, string memory version) public {
  ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenLinearBondedEthDeployer.sol:583:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor (string memory name_, string memory symbol_) public {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenLinearBondedEthDeployer.sol:910:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor () internal {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenLinearBondedEthDeployer.sol:2723:5: Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
    constructor () internal {
    ^ (Relevant source part starts here and spans across multiple lines).

contracts/flatten/TokenLinearBondedEthDeployer.sol:2667:5: Warning: Unused local variable.
    uint256 m = slope;
    ^-------^

Compilation finished successfully
·-----------------------------------|-------------·
|  Contract Name                    ·  Size (Kb)  |
····································|·············
|  Address                          ·       0.08  |
····································|·············
|  Address                          ·       0.08  |
····································|·············
|  Address                          ·       0.08  |
····································|·············
|  Address                          ·       0.08  |
····································|·············
|  AddressUpgradeable               ·       0.08  |
····································|·············
|  AddressUpgradeable               ·       0.08  |
····································|·············
|  AddressUpgradeable               ·       0.08  |
····································|·············
|  AddressUpgradeable               ·       0.08  |
····································|·············
|  Coinvise                         ·      14.05  |
····································|·············
|  console                          ·       0.08  |
····································|·············
|  console                          ·       0.08  |
····································|·············
|  console                          ·       0.08  |
····································|·············
|  console                          ·       0.08  |
····································|·············
|  console                          ·       0.08  |
····································|·············
|  EIP712Base                       ·       0.06  |
····································|·············
|  EIP712Base                       ·       0.06  |
····································|·············
|  EIP712Base                       ·       0.06  |
····································|·············
|  EIP712BaseUpgradeable            ·       0.80  |
····································|·············
|  EIP712BaseUpgradeable            ·       0.80  |
····································|·············
|  EIP712BaseUpgradeable            ·       0.80  |
····································|·············
|  EIP712MetaTransaction            ·       2.11  |
····································|·············
|  EIP712MetaTransaction            ·       2.11  |
····································|·············
|  EIP712MetaTransaction            ·       2.04  |
····································|·············
|  EIP712MetaTransactionUpgradeable ·       2.79  |
····································|·············
|  EIP712MetaTransactionUpgradeable ·       2.79  |
····································|·············
|  EIP712MetaTransactionUpgradeable ·       2.79  |
····································|·············
|  EnumerableMap                    ·       0.08  |
····································|·············
|  EnumerableSet                    ·       0.08  |
····································|·············
|  EnumerableSet                    ·       0.08  |
····································|·············
|  EnumerableSet                    ·       0.08  |
····································|·············
|  ERC20                            ·       2.49  |
····································|·············
|  ERC20                            ·       2.49  |
····································|·············
|  ERC20                            ·       2.49  |
····································|·············
|  ERC20                            ·       2.49  |
····································|·············
|  ERC20Token                       ·       4.07  |
····································|·············
|  ERC20Token                       ·       4.07  |
····································|·············
|  ERC20Token                       ·       4.07  |
····································|·············
|  ERC20Token                       ·       4.07  |
····································|·············
|  Erc20TokenLinearBoundedEthPayable ·      5.43  |
····································|·············
|  Erc20TokenLinearBoundedEthPayable ·      5.43  |
····································|·············
|  ERC721                           ·       6.08  |
····································|·············
|  ERC721Token                      ·      10.31  |
····································|·············
|  NFTAirdrop                       ·       9.23  |
····································|·············
|  NFTMarketplace                   ·      13.53  |
····································|·············
|  NFTProxy                         ·       4.15  |
····································|·············
|  SafeERC20                        ·       0.08  |
····································|·············
|  SafeERC20                        ·       0.08  |
····································|·············
|  SafeERC20                        ·       0.08  |
····································|·············
|  SafeMath                         ·       0.08  |
····································|·············
|  SafeMath                         ·       0.08  |
····································|·············
|  SafeMath                         ·       0.08  |
····································|·············
|  SafeMath                         ·       0.08  |
····································|·············
|  SafeMath                         ·       0.08  |
····································|·············
|  SafeMath                         ·       0.08  |
```

```
·····································|·············
|  SafeMath                          ·       0.08  |
·····································|·············
|  SafeMath                          ·       0.08  |
·····································|·············
|  SafeMath                          ·       0.08  |
·····································|·············
|  Strings                           ·       0.08  |
·····································|·············
|  TokenDeployer                     ·       9.24  |
·····································|·············
|  TokenLinearBondedEthDeployer      ·      10.63  |
·····································|·············
You're running a network fork starting from the latest block.
Performance may degrade due to fetching data from the network with each run.
If connecting to an archival node (e.g. Alchemy), we strongly recommend setting
blockNumber to a fixed value to increase performance with a local cache.

  Deposit

    1) "before each" hook for "should deposit token"

  Coinvise

    2) "before each" hook for "Estimate Gas"

  ERC20Token

    3) "before each" hook for "Should transfer tokens"

  ERC721Token
     ✓ Should mint NFTs

  EthPool

    4) "before each" hook for "Should recieve ETH properly"

  Linear Bonding Curve Erc20 Payable

    5) "before each" hook for "Should mint tokens on a linear bonding curve"

  Linear Bonding Curve ETH Payable

    6) "before each" hook for "Should mint tokens on a linear bonding curve"

  NFTAirdrop Test
    Gas Estimation
Gas estimate for 100 links: 12137063. Assuming gas price of 250 gwei, eth price = 3.03426575
       ✓ Estimate Gas: Airdrop with 100 Links
    Functionality
      Campaign Creation
         ✓ Should Create Campaign
         ✓ Should revert if all are not approved
         ✓ Should revert if fee is not paid
         ✓ Should revert if mismatch between id list and links count
      Claim
         ✓ Users should be able to claim
         ✓ Should revert if claim happens after links are exhaused
         ✓ Users should be able to claim only once
         ✓ Should revert if signature is invalid

  NFTMarketplace
    Functionality
      Listing

        7) "before each" hook for "Should be able to list NFT payable by ETH"
    Payment Token Whitelisting

      8) "before each" hook for "Should revert if Listing is attempted using non whitelisted token"
    Upgradeability

      9) Should be upgradeable

  NFTProxy
    Functionality
      Listing

        10) Should be able to list NFT payable by ETH

        11) Owner should be able to de-list a nft before it's bought

        12) Only listing owner should be able to delist
      Buying with ETH

        13) "before each" hook for "User should be able to buy NFT using ETH"
      NFTProxy Pausibility
         ✓ Should revert if non owner tries to change state

        14) Should revert if NFT is listed in paused proxy

        15) Should revert if NFT is bought in paused proxy

  Sigmoid Bonding Curve Erc20 Payable

    16) "before each" hook for "Should Calculate buy price properly"

  Sigmoid Bonding Curve ETH Payable

    17) "before each" hook for "Should Calculate buy price properly"

  TokenDeployer

    18) Should deploy Erc20Token from MetaTransaction

·····································|··························|·············|·····························
|         Solc version: 0.7.4         ·  Optimizer enabled: true  ·  Runs: 200  ·  Block limit: 12500000 gas  |
·····································|··························|·············|·····························
|  Methods                                                                                                   |
·················|···················|·············|·············|·············|··············|··············
|  Contract      ·  Method           ·  Min        ·  Max        ·  Avg        ·  # calls     ·  eur (avg)  |
·················|···················|·············|·············|·············|··············|··············
|  Coinvise      ·  claim            ·     135639  ·     159039  ·     149950  ·          14  ·          -  |
·················|···················|·············|·············|·············|··············|··············
|  ERC721Token   ·  mint             ·     167085  ·     197085  ·     169049  ·         168  ·          -  |
·················|···················|·············|·············|·············|··············|··············
|  ERC721Token   ·  setApprovalForAll  ·      45213  ·      45225  ·      45221  ·           8  ·          -  |
·················|···················|·············|·············|·············|··············|··············
|  NFTAirdrop    ·  createCampaign   ·     528889  ·     624277  ·     560685  ·           6  ·          -  |
·················|···················|·············|·············|·············|··············|··············
|  NFTProxy      ·  pauseProxy       ·          -  ·          -  ·      45766  ·           1  ·          -  |
·················|···················|·············|·············|·············|··············|··············
|  Deployments                       ·                                        ·  % of limit   ·             |
·····································|·············|·············|·············|··············|··············
|  ERC721Token                       ·    2546530  ·    2546578  ·    2546535  ·       20.4 %  ·          -  |
·····································|·············|·············|·············|··············|··············
|  NFTAirdrop                        ·          -  ·          -  ·    2093458  ·       16.7 %  ·          -  |
·····································|·············|·············|·············|··············|··············
|  NFTProxy                          ·          -  ·          -  ·     970931  ·        7.8 %  ·          -  |
·····································|·············|·············|·············|··············|··············
|  TokenDeployer                     ·          -  ·          -  ·    2144301  ·       17.2 %  ·          -  |
·····································|·············|·············|·············|··············|··············

  11 passing (33s)
  18 failing

  1) Deposit
       "before each" hook for "should deposit token":
     HardhatError: HH700: Artifact for contract "Dai" not found.
      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/Deposit.test.ts:24:17)

  2) Coinvise
       "before each" hook for "Estimate Gas":
     HardhatError: HH700: Artifact for contract "Dai" not found.
      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/Distribution.test.ts:43:17)

  3) ERC20Token
       "before each" hook for "Should transfer tokens":
     HardhatError: HH701: There are multiple artifacts for contract "ERC20Token", please use a fully qualified name.

Please replace ERC20Token for one of these options wherever you are trying to read its artifact:

contracts/flatten/Erc20Token.sol:ERC20Token
contracts/flatten/Erc20TokenLinearBondedEthPayable.sol:ERC20Token
contracts/flatten/TokenDeployer.sol:ERC20Token
contracts/flatten/TokenLinearBondedEthDeployer.sol:ERC20Token

      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:401:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
```

```
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/ERC20Token.tests.ts:17:29)

  4) EthPool
       "before each" hook for "Should recieve ETH properly":
     HardhatError: HH700: Artifact for contract "EthPool" not found.
      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/EthPool.test.ts:17:29)

  5) Linear Bonding Curve Erc20 Payable
       "before each" hook for "Should mint tokens on a linear bonding curve":
     HardhatError: HH700: Artifact for contract "Dai" not found.
      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/LinearBondingCurve.Erc20Payable.test.ts:19:17)

  6) Linear Bonding Curve ETH Payable
       "before each" hook for "Should mint tokens on a linear bonding curve":
     HardhatError: HH701: There are multiple artifacts for contract "Erc20TokenLinearBoundedEthPayable", please use a fully qualified name.

Please replace Erc20TokenLinearBoundedEthPayable for one of these options wherever you are trying to read its artifact:

contracts/flatten/Erc20TokenLinearBondedEthPayable.sol:Erc20TokenLinearBoundedEthPayable
contracts/flatten/TokenLinearBondedEthDeployer.sol:Erc20TokenLinearBoundedEthPayable

      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:401:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/LinearBondingCurve.EthPayable.test.ts:19:29)

  7) NFTMarketplace
       Functionality
         "before each" hook for "Should be able to list NFT payable by ETH":
     HardhatError: HH700: Artifact for contract "Dai" not found.
      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/NFTMarketplace.test.ts:23:34)

  8) NFTMarketplace
       Payment Token Whitelisting
         "before each" hook for "Should revert if Listing is attempted using non whitelisted token":
     HardhatError: HH700: Artifact for contract "Dai" not found.
      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/NFTMarketplace.test.ts:650:34)

  9) NFTMarketplace
       Upgradeability
         Should be upgradeable:
     HardhatError: HH700: Artifact for contract "NFTMarketplaceDud" not found.
      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/NFTMarketplace.test.ts:754:48)

  10) NFTProxy
       Functionality
         Listing
           Should be able to list NFT payable by ETH:
     Error: missing argument: passed to contract (count=3, expectedCount=5, code=MISSING_ARGUMENT, version=contracts/5.0.12)
      at Logger.makeError (node_modules/@ethersproject/logger/src.ts/index.ts:205:28)
      at Logger.throwError (node_modules/@ethersproject/logger/src.ts/index.ts:217:20)
      at Logger.checkArgumentCount (node_modules/@ethersproject/logger/src.ts/index.ts:276:18)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/src.ts/index.ts:159:12
      at step (node_modules/@ethersproject/contracts/lib/index.js:48:23)
      at Object.next (node_modules/@ethersproject/contracts/lib/index.js:29:53)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/lib/index.js:23:71
      at new Promise (<anonymous>)
      at __awaiter (node_modules/@ethersproject/contracts/lib/index.js:19:12)
      at populateTransaction (node_modules/@ethersproject/contracts/lib/index.js:138:12)

  11) NFTProxy
       Functionality
         Listing
           Owner should be able to de-list a nft before it's bought:
     Error: missing argument: passed to contract (count=3, expectedCount=5, code=MISSING_ARGUMENT, version=contracts/5.0.12)
      at Logger.makeError (node_modules/@ethersproject/logger/src.ts/index.ts:205:28)
      at Logger.throwError (node_modules/@ethersproject/logger/src.ts/index.ts:217:20)
      at Logger.checkArgumentCount (node_modules/@ethersproject/logger/src.ts/index.ts:276:18)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/src.ts/index.ts:159:12
      at step (node_modules/@ethersproject/contracts/lib/index.js:48:23)
      at Object.next (node_modules/@ethersproject/contracts/lib/index.js:29:53)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/lib/index.js:23:71
      at new Promise (<anonymous>)
      at __awaiter (node_modules/@ethersproject/contracts/lib/index.js:19:12)
      at populateTransaction (node_modules/@ethersproject/contracts/lib/index.js:138:12)

  12) NFTProxy
       Functionality
         Listing
           Only listing owner should be able to delist:
     Error: missing argument: passed to contract (count=3, expectedCount=5, code=MISSING_ARGUMENT, version=contracts/5.0.12)
      at Logger.makeError (node_modules/@ethersproject/logger/src.ts/index.ts:205:28)
      at Logger.throwError (node_modules/@ethersproject/logger/src.ts/index.ts:217:20)
      at Logger.checkArgumentCount (node_modules/@ethersproject/logger/src.ts/index.ts:276:18)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/src.ts/index.ts:159:12
      at step (node_modules/@ethersproject/contracts/lib/index.js:48:23)
      at Object.next (node_modules/@ethersproject/contracts/lib/index.js:29:53)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/lib/index.js:23:71
      at new Promise (<anonymous>)
      at __awaiter (node_modules/@ethersproject/contracts/lib/index.js:19:12)
      at populateTransaction (node_modules/@ethersproject/contracts/lib/index.js:138:12)

  13) NFTProxy
       Functionality
         Buying with ETH
           "before each" hook for "User should be able to buy NFT using ETH":
     Error: missing argument: passed to contract (count=3, expectedCount=5, code=MISSING_ARGUMENT, version=contracts/5.0.12)
      at Logger.makeError (node_modules/@ethersproject/logger/src.ts/index.ts:205:28)
      at Logger.throwError (node_modules/@ethersproject/logger/src.ts/index.ts:217:20)
      at Logger.checkArgumentCount (node_modules/@ethersproject/logger/src.ts/index.ts:276:18)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/src.ts/index.ts:159:12
      at step (node_modules/@ethersproject/contracts/lib/index.js:48:23)
      at Object.next (node_modules/@ethersproject/contracts/lib/index.js:29:53)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/lib/index.js:23:71
      at new Promise (<anonymous>)
      at __awaiter (node_modules/@ethersproject/contracts/lib/index.js:19:12)
      at populateTransaction (node_modules/@ethersproject/contracts/lib/index.js:138:12)

  14) NFTProxy
       Functionality
         NFTProxy Pausibility
           Should revert if NFT is listed in paused proxy:
      AssertionError: Expected transaction to be reverted with PROXY_PAUSED, but other exception was thrown: Error: missing argument: passed to contract (count=3, expectedCount=5, code=MISSING_ARGUMENT,
version=contracts/5.0.12)


  15) NFTProxy
       Functionality
         NFTProxy Pausibility
           Should revert if NFT is bought in paused proxy:
     Error: missing argument: passed to contract (count=3, expectedCount=5, code=MISSING_ARGUMENT, version=contracts/5.0.12)
      at Logger.makeError (node_modules/@ethersproject/logger/src.ts/index.ts:205:28)
      at Logger.throwError (node_modules/@ethersproject/logger/src.ts/index.ts:217:20)
      at Logger.checkArgumentCount (node_modules/@ethersproject/logger/src.ts/index.ts:276:18)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/src.ts/index.ts:159:12
      at step (node_modules/@ethersproject/contracts/lib/index.js:48:23)
      at Object.next (node_modules/@ethersproject/contracts/lib/index.js:29:53)
      at /Users/jose/lab/audits/CoinVise/coinvise-contracts-b5f5c1f316e5e3545ed5a712463d2e85f0e76f0e/node_modules/@ethersproject/contracts/lib/index.js:23:71
      at new Promise (<anonymous>)
      at __awaiter (node_modules/@ethersproject/contracts/lib/index.js:19:12)
      at populateTransaction (node_modules/@ethersproject/contracts/lib/index.js:138:12)

  16) Sigmoid Bonding Curve Erc20 Payable
       "before each" hook for "Should Calculate buy price properly":
     HardhatError: HH700: Artifact for contract "Dai" not found.
      at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
      at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
      at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
      at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
      at Context.<anonymous> (test/SigmoidBondingCurve.Erc20Payable.test.ts:21:17)

  17) Sigmoid Bonding Curve ETH Payable
       "before each" hook for "Should Calculate buy price properly":
```

```
    HardhatError: HH700: Artifact for contract "Erc20TokenSigmoidBoundedEthPayable" not found.
     at Artifacts._getArtifactPathFromFiles (node_modules/hardhat/src/internal/artifacts.ts:391:13)
     at Artifacts._getArtifactPath (node_modules/hardhat/src/internal/artifacts.ts:316:17)
     at Artifacts.readArtifact (node_modules/hardhat/src/internal/artifacts.ts:49:26)
     at getContractFactoryByName (node_modules/@nomiclabs/hardhat-ethers/src/internal/helpers.ts:100:20)
     at Context.<anonymous> (test/SigmoidBondingCurve.EthPayable.test.ts:21:29)

  18) TokenDeployer
       Should deploy Erc20Token from MetaTransaction:
     Error: types/values length mismatch (count={"types":5,"values":4}, value={"types":
[{"name":"name","type":"string","indexed":null,"components":null,"arrayLength":null,"arrayChildren":null,"baseType":"string","_isParamType":true},
{"name":"symbol","type":"string","indexed":null,"components":null,"arrayLength":null,"arrayChildren":null,"baseType":"string","_isParamType":true},
{"name":"minter","type":"address","indexed":null,"components":null,"arrayLength":null,"arrayChildren":null,"baseType":"address","_isParamType":true},
{"name":"supplyCap","type":"uint256","indexed":null,"components":null,"arrayLength":null,"arrayChildren":null,"baseType":"uint256","_isParamType":true},
{"name":"integrationFee","type":"uint256","indexed":null,"components":null,"arrayLength":null,"arrayChildren":null,"baseType":"uint256","_isParamType":true}],"values":
["Test","$TEST","0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266",1000000]}, code=INVALID_ARGUMENT, version=abi/5.0.13)
      at Logger.makeError (node_modules/@ethersproject/logger/src.ts/index.ts:205:28)
      at Logger.throwError (node_modules/@ethersproject/logger/src.ts/index.ts:217:20)
      at AbiCoder.encode (node_modules/@ethersproject/abi/src.ts/abi-coder.ts:102:20)
      at Interface._encodeParams (node_modules/@ethersproject/abi/src.ts/interface.ts:267:31)
      at Interface.encodeFunctionData (node_modules/@ethersproject/abi/src.ts/interface.ts:297:18)
      at Context.<anonymous> (test/TokenDeployer.test.ts:43:63)
```

## Code Coverage

Code coverage reports were generated using Hardhat plugin. Test file `test/NFTAirdrop.test.ts` was excluded from the coverage run to avoid an error on this part of the testing suite. We leave the original Code Coverage Report from the initial audit. As for the Reaudit, we found [that] the module could not be supported without several fixes on the project configuration.

| File | % Stmts | % Branch | % Funcs | % Lines | Uncovered Lines |
|---|---|---|---|---|---|
| **contracts/** | 70.79 | 52.17 | 56.63 | 71.61 | |
| Coinvise.sol | 88.24 | 63.04 | 61.54 | 88.24 | … 573,577,578 |
| Dai.sol | 50 | 26.92 | 53.33 | 53.85 | … 121,122,123 |
| EthPool.sol | 66.67 | 100 | 50 | 71.43 | 24,39 |
| NFTAirdrop.sol | 0 | 0 | 0 | 0 | … 285,287,295 |
| NFTMarketplace.sol | 94.79 | 77.27 | 83.33 | 94.9 | … 146,154,155 |
| TokenDeployer.sol | 100 | 100 | 100 | 100 | |
| **contracts/lib/** | 50 | 30 | 64.71 | 48.94 | |
| BasicMetaTransaction.sol | 0 | 0 | 0 | 0 | … 106,107,115 |
| EIP712Base.sol | 100 | 100 | 100 | 100 | |
| EIP712MetaTransaction.sol | 85.71 | 50 | 100 | 82.61 | 44,124,125,126 |
| **contracts/lib/EIP712MetaTransactionUpgradeable/** | 96 | 66.67 | 100 | 96.43 | |
| EIP712BaseUpgradeable.sol | 100 | 100 | 100 | 100 | |
| EIP712MetaTransactionUpgradeable.sol | 95.45 | 66.67 | 100 | 95.83 | 54 |
| **contracts/testContracts/** | 100 | 100 | 100 | 100 | |
| NFTMarketplaceDud.sol | 100 | 100 | 100 | 100 | |
| **contracts/tokens/** | 29.03 | 17.86 | 28.57 | 29.03 | |
| Erc20Token.sol | 90.91 | 83.33 | 83.33 | 90.91 | 30 |
| Erc721Token.sol | 100 | 100 | 100 | 100 | |
| EthereumToken.sol | 0 | 0 | 0 | 0 | … 54,58,65,66 |
| MaticToken.sol | 0 | 0 | 0 | 0 | … 122,131,132 |
| **All files** | **64.41** | **45.96** | **55.4** | **64.98** | |

# 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

```
dce81dc2414e3fc314e381fd3eff87c0caef5f88442970e81effb432311dcf83 ./contracts/Dai.sol
fc7e9c0c7242ed496c106e64be1cadee12e331e08831e90461af8190c8b3ba4a ./contracts/Coinvise.sol
9d797d20e03736cd360668e0382de427fc52f55dcd45c7dfa9c3794ac7248a18 ./contracts/TokenLinearBondedEthDeployer.sol
46401667e4837192a65a2bc399d35bb04cd13dee38b5e6ac33f7c17b3eefc922 ./contracts/EthPool.sol
5f2cfc7a35a7161dfee38551cf61207a51445e9b07f5b7b794a9b441c9167559 ./contracts/NFTAirdrop.sol
802ab4dc5a5e67a7242b799e6289e1198877b883d6c8f54ebe9dbdecdc4fa104 ./contracts/NFTProxy.sol
ab65d62e81cdd73b554bd4157a00bcd08266dc16ff5484352f51ba0a6d9b9358 ./contracts/NFTMarketplace.sol
ec11a56924ffb7b1578a06e4eaa4ce38ca8f39810d51fb0874d26585a9c55642 ./contracts/TokenDeployer.sol
18aa6e96b8b9555de363b2e5a86f82b404c5b261eeec9fcf9ae8fbf51dd35784 ./contracts/tokens/Erc721Token.sol
b1cefd8f12e3e5d27a102a26d5cc1607e79af2a90429bf5378512ebcd158ecd3 ./contracts/tokens/Erc20Token.sol
b2d08281e32cccaf1664eea571d395e15389a6048bf4b68de3aa9e8790b35df9 ./contracts/tokens/Erc20TokenLinearBondedErc20Payable.sol
9ef14aaa2355582ecea2614a9e51d8ea0235d986362e13ec0b9fc0958f8ba5c5 ./contracts/tokens/MaticToken.sol
73f624f3c6bffccb5d2014fd8c9ebde4521cfff6835229a17e9a9b70ce91c7af ./contracts/tokens/Erc20TokenSigmoidBondedErc20Payable.sol
1992ff918906c9715e6f3ecde4f5981aaca0309631cbe042aa0358bc0faaa844 ./contracts/tokens/Erc20TokenLinearBondedEthPayable.sol
01f07d0594c52c6b60eae7447f380ab502e5f6b0b09eedebc32ef11b2da5b5e6 ./contracts/tokens/EthereumToken.sol
f6d53dd2346f633b63d271417c1c565131711d10479618e3d2d58c37ad1c492d ./contracts/tokens/Erc20TokenSigmoidBondedEthPayable.sol
7de5f33b1b19e413321d5c1c1bcb13296beb32c4d9e37bb6b9996af01e9a05e2 ./contracts/lib/EIP712Base.sol
d622f6513afa90eff7c217cc8e4b8e67c996868d6b0d53c75c70f3cd64369665 ./contracts/lib/BasicMetaTransaction.sol
50842de7f95080173a89e541c0b44262fddf415e94ab2726573fe5258f8c9f7f ./contracts/lib/EIP712MetaTransaction.sol
b9e6a7f83e67fa745417761105dd4a0b9e06f05caa3c76cade657684334d6483 ./contracts/lib/EIP712MetaTransactionUpgradeable/EIP712BaseUpgradeable.sol
da7664be6198aff3da13a0c60d7e7a685589c72f782f50cc640b0ce7add64d91
./contracts/lib/EIP712MetaTransactionUpgradeable/EIP712MetaTransactionUpgradeable.sol
c1da0f1d9f2dd1c32b6bed0a720cf51054bb3179666a5bd73c1dd80387847c82 ./contracts/flatten/Erc721Token.sol
2f29fb76473f55cd4bb852b2ffce27473d0e4d21d6257dbfc562395137eb0bde ./contracts/flatten/Coinvise.sol
9641b506044f0feb93ce0fc4f3b0f547ef79015922c55618ca0c843b80fd6b51 ./contracts/flatten/Erc20Token.sol
43b1eba3f4661246c8d4540f7668bf124c033dac2610775b8ae6dad1643a51c9 ./contracts/flatten/TokenLinearBondedEthDeployer.sol
40ac84521043c7999c7dfa1dbd035b65c645102ea4d8011dd10bd930c1be8fdf ./contracts/flatten/NFTAirdrop.sol
d37668db30903998cd1113945b1dea2781292ef561ef1b07f75ee7fcd31cc303 ./contracts/flatten/NFTProxy.sol
d8bbd0813bb5565131e91e128db66d0430694e5d862948334522e4145eea408a ./contracts/flatten/NFTMarketplace.sol
6fad0c5307caa51cb5cd66ddcdadeba151c9dfd24340acbd4c1c7b1d284afb99 ./contracts/flatten/TokenDeployer.sol
8645183c092073d2cade0725ee928b2f5546c18f26dab847cae55d389328dc27 ./contracts/flatten/Erc20TokenLinearBondedEthPayable.sol
```

### Tests

```
ee1203c4bab537d7bf264eb90ddacc015317a2310be8f0cdc39612029752706f ./test/NFTProxy.test.ts
786bd65e5e07ebd2d68bb8a80dc43eb6ee099fa73a864fef7e9c352c52a39ceb ./test/LinearBondingCurve.EthPayable.test.ts
42e8c05cd720cf557360118e16648c72a4f903b5596c8ccbc9c2d80b48bdca13 ./test/NFTAirdrop.test.ts
efb9c60e4046f284dcc5bcf020dbe8ca6ed10f3c6f615ed0e6372242d8abc21b ./test/ERC20Token.tests.ts
e9191c3165dda3d1e85d50c8aa2cce9603407b086fe36995c614a3e97acd0f88 ./test/ERC721Token.test.ts
9e36590746fcd87cf131505ff061bae9c7f5bc4ee1fdd4ae78b38e6fb83366b8 ./test/TokenDeployer.test.ts
26f45f37412f0ae481c12775744cfdbfe66ef8a47f0536a8a9a2e8184c70987f ./test/SigmoidBondingCurve.Erc20Payable.test.ts
188d0ae8e0d3118678716b95420a2480af8671ffcb98ead886c5cd7e92ebac27 ./test/EthPool.test.ts
e2ba0a7effc1291755b4d2bf57f7e3f8129590cdc7ac1ab2284db17debf9d288 ./test/LinearBondingCurve.Erc20Payable.test.ts
3b227308e1b27a24dfd979981a94595a56de6bd4a70c6d06466d1b7569206f55 ./test/Deposit.test.ts
70e4d78af3cae9ff38d366ceaaeda8f65ee53eaeff947151e3ce607756f416f9 ./test/test-utils.ts
98144985eaeeb07df916f8a5dc35d19c1ece4f4b7ba86f7ce20bc392dc242440 ./test/SigmoidBondingCurve.EthPayable.test.ts
5bd3f628baabf802717f5c23a3cefa3c06486a4c221128430de99d56f930d646 ./test/NFTMarketplace.test.ts
86ad975c1cb7ce3ba3e4a9bf53a5e3b0cd0de5be7fd0912c8f701eea5f81db0b ./test/Distribution.test.ts
```

# Changelog

- 2021-05-25 - Initial report
- 2021-06-15 - Reaudit report

## 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.