SOLIDIFIED

Audit Report for Etched - June 21, 2021

## Summary

Audit Report prepared by Solidified covering the Etched smart contracts.

## Process and Delivery

Three (3) independent Solidified experts performed an unbiased and isolated audit of the code. The debrief on 21 June 2021.

## Audited Files

The source code has been supplied in the form of a GitLab repository:

https://gitlab.com/linumlabs/etched/-/tree/audit/von/audit-branch/chain

Commit number: `2af64f885eb84ae62c9a147d2bb800460fa54baf`

The scope of the audit was limited to the following files:

```
contracts
├── Royalties
│   ├── IRoyalties.sol
│   └── Royalties.sol
├── auctions
│   ├── AuctionHub.sol
│   ├── BaseAuction.sol
│   ├── IAuction.sol
│   ├── IHub.sol
│   ├── PrivateAuction.sol
│   ├── first-price
│   │   ├── FirstPrice.sol
│   │   └── PrivateFirstPrice.sol
│   ├── single-price
│   │   ├── PrivateSinglePrice.sol
│   │   └── SinglePrice.sol
│   └── timed-edition
│       ├── BaseEdition.sol
│       ├── BasePrivateEdition.sol
│       ├── EditionData.sol
│       ├── limited
│       │   ├── LimitedTimedEdition.sol
│       │   └── PrivateLimitedTimedEdition.sol
│       └── unlimited
│           ├── PrivateTimedEdition.sol
│           └── TimedEdition.sol
├── nft
│   ├── INft.sol
```

```
│   ├── NFT.sol
│   └── NftBase.sol
└── registry
    └── Registry.sol
```

## Intended Behavior

The smart contracts implement the smart contracts for an NFT marketplace with several auction models.

## Code Complexity and Test Coverage

Smart contract audits are an important step to improve the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of a smart contract system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**.

**Note, that high complexity or lower test coverage does equate to a higher risk. Certain bugs are more easily detected in unit testing than a security audit and vice versa. It is, therefore, more likely that undetected issues remain if the test coverage is low or non-existent.**

| Criteria | Status | Comment |
|---|---|---|
| Code complexity | Medium | - |
| Code readability and clarity | High | - |
| Level of Documentation | High | - |
| Test Coverage | High | - |

SOLIDIFIED

Audit Report for Etched - June 21, 2021

## Issues Found

Solidified found that the Etched contracts contain no critical issues, 2 major issues, 3 minor issues, in addition to 4 informational notes.

We recommend all issues are amended, while the notes are up to the team's discretion, as they refer to best practices.

| Issue # | Description | Severity | Status |
|---|---|---|---|
| 1 | Royalties.sol: Missing validations can cause loss of funds | Major | Pending |
| 2 | AuctionHub.sol: Auction with active lots can be removed or updated | Major | Pending |
| 3 | Casing error in import stops hardhat compilation | Minor | Pending |
| 4 | NFTBase.sol: Burn does not decrease circulating supply | Minor | Pending |
| 5 | Compiler Version | Note | - |
| 6 | Consider using pull over push for payment | Note | - |
| 7 | Code cleanup | Note | - |
| 8 | NFT.sol: tokenDrain() will fail if an ERC20 token's transfer() function does not return true | Note | - |
| 9 | NFT.sol: Unnecessary state variable write operation | Note | - |

# Critical Issues

No critical issues have been found.

# Major Issues

## 1. Royalties.sol: Missing validations can cause loss of funds

The function `updateAddress()` is missing a few important validations. The method never checks if the `_newAddress` already has any existing royalty. Overwriting an existing address with royalty will make the funds to be lost forever and can never be withdrawn.

Furthermore, the method also allows the owner to use any address as input which gives the owner complete control over any account balance stored in the contract.

**Recommendation**
It is recommended to sum the existing and new royalty together and store rather than completely overwriting the existing royalty balance.

The method acts more like a backup plan but can be easily used by malicious owners to claim all royalty. It is recommended to either remove this method or inform the user beforehand if it's offered as a feature.

## 2. AuctionHub.sol: Auction with active lots can be removed or updated

The function `removeAuction()` and `updateAuction()` do not check if there are any existing active Lots. Any such removed auction will lock the token transferred to it permanently.

**Recommendation**
Consider checking if there are any lots already present in the auction before updating or removing it from the auction hub.

# Minor Issues

## 3. Casing error in import stops hardhat compilation

In the following files the directory Royalties is spelled with a minor ‘r’ instead of capital ‘R’ in the import statements:

`BaseAcution.sol`
`Registry.sol`

This prevents the hardhat build script from executing, since it is case sensitive.

**Recommendation**
Clean up casing to make hardhat build work.

## 4. NFTBase.sol: Burn does not decrease circulating supply

The function `_burn()` does not reduce the circulating supply count.

**Recommendation**
It is recommended to decrease the circulating supply when a token is burnt.

# Informational Notes

## 5. Compiler Version

The codebase locks the compiler version to 0.7.3. In general it is good practise to lock the version pragma to a specific version. However, Solidity version 0.7.3 contained several compiler bugs, including a security relevant code generation bug that can lead to data corruption when copying empty arrays to storage.

**Recommendation**
Consider using compiler version 0.7.4 or 0.7.5

## 6. Consider using pull over push for payment

The function `_insecureHandlePayment()` sends the sale amount to the seller in the same method. This allows the seller to annoy/manipulate the system to some extent. For example, the seller can maintain a whitelist of their own or make the buyer pay more gas.

**Recommendation**
Consider using the existing royalty module to pay the seller as well.

## 7. Code cleanup

Consider cleaning up the code based on the following recommendations.

1. `PrivateAuction.sol`: Consider using an address set for whitelisting to prevent any duplicates and ease of use.
2. Remove the `Testable.sol` import from all contracts before deploying to the main net.
3. `SinglePrice.sol`: The value of biddable will never be set to false by the method `_isLotBiddable` since the method calling it will always throw for false input.
4. `BaseEdition.sol`: Includes unreachable code in `_isValidCreator` method. The `else-if` path of the condition statement is redundant, because it will never be executed.
5. `BaseAuction.sol`: Function `_isLotInBiddableState` the condition `status != IHub.LotStatus.AUCTION_CANCELED` is redundant.
6. `AuctionHub.sol`: misspelled `actionAddress_` contract variable

## 8. NFT.sol: tokenDrain() will fail if an ERC20 token's transfer() function does not return true

The function `tokenDrain()` relies on the provided token to return `true` on successful transfer. However, this is not always the case. Some tokens may succeed silently. In this case, the function would always revert.

**Recommendation**
Consider either using SafeERC20 or remove the `require` check token transfer return value.

## 9. NFT.sol: Unnecessary state variable write operation

The `onlyAuctions` modifier sets the `auctionHubInstance_ contract` variable every time it is called. However, the variable is never read.

**Recommendation**
Consider removing the unnecessary variable and write operation.

## Disclaimer

Solidified audit is not a security warranty, investment advice, or an endorsement of Etched or its products. This audit does not provide a security or correctness guarantee of the audited smart contract. Securing smart contracts is a multistep process, therefore running a bug bounty program as a complement to this audit is strongly recommended.

The individual audit reports are anonymized and combined during a debrief process, in order to provide an unbiased delivery and protect the auditors of Solidified platform from legal and financial liability.

*Solidified Technologies Inc.*