Quantstamp Contract Security Certificate

February 3rd 2020 — Quantstamp Verified

AirSwap

This smart contract audit was prepared by Quantstamp, the protocol for securing smart contracts.

Executive Summary

Type	Peer-to-Peer Trading Smart Contracts
Auditors	Ed Zulkoski, Senior Security Engineer Kacper Bąk, Senior Research Engineer Sung-Shine Lee, Research Engineer
Timeline	2019-11-04 through 2019-12-20
EVM	Constantinople
Languages	Solidity, Javascript
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	AirSwap Documentation
Source Code	Repository: airswap-protocols — Commit: b87d292

Goals

- Can funds be locked in the contracts?
- Are funds properly exchanged when a swap occurs?
- Do the contracts adhere to Solidity best practices?

Changelog

- 2019-11-20 - Initial report
- 2019-11-26 - Revised report based on commit bdf1289
- 2019-12-04 - Revised report based on commit 8798982
- 2019-12-04 - Revised report based on commit f161d31
- 2019-12-20 - Revised report based on commit 5e8a07c
- 2020-01-20 - Revised report based on commit 857e296

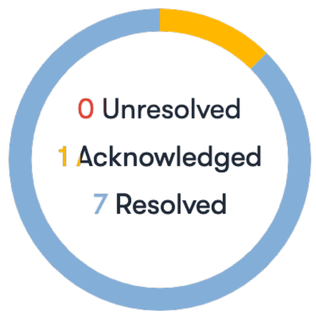
Overall Assessment

The AirSwap smart contracts are well-documented and generally follow best practices. However, several issues were discovered during the audit that may cause the contracts to not behave as intended, such as funds being to be locked in contracts, or incorrect checks on external contract calls. These findings, along with several other issues noted below, should be addressed before the contracts are ready for production.

Update: Fluidity has addressed our concerns as of commit 857e296.

Disclaimer: as the contracts in `source/tokens/contracts/` are claimed to be direct copies from OpenZeppelin or deployed contracts taken from Etherscan, with minor event/variable name changes. These files were not included as part of the final audit.

Total Issues	**9** (7 Resolved)
High Risk Issues	**1** (1 Resolved)
Medium Risk Issues	**2** (2 Resolved)
Low Risk Issues	**4** (3 Resolved)
Informational Risk Issues	**2** (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).
Resolved	Adjusted program implementation, requirements or constraints to eliminate the risk.

Summary of Findings

ID	Description	Severity	Status
QSP-1	Funds may be locked if `setRuleAndIntent` is called multiple times	High	Resolved
QSP-2	Centralization of Power	Medium	Resolved
QSP-3	Integer arithmetic may cause incorrect pricing logic	Medium	Resolved
QSP-4	`transferFrom()` success should not be checked by querying token balances	Low	Resolved
QSP-5	`isValid()` does not check that the `validator` contract is correct	Low	Resolved
QSP-6	Unchecked Return Value	Low	Resolved
QSP-7	Gas Usage / `for` Loop Concerns	Low	Acknowledged
QSP-8	Return values of ERC20 function calls are not checked	Informational	Resolved
QSP-9	Unchecked constructor argument	Informational	-

Quantstamp Audit Breakdown

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

Toolset

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

Setup

Tool Setup:

- Maian
- Truffle
- Ganache
- SolidityCoverage
- Mythril
- Truffle-Flattener
- Securify
- Slither

Steps taken to run the tools:

1. Installed Truffle: `npm install -g truffle`
2. Installed Ganache: `npm install -g ganache-cli`
3. Installed the solidity-coverage tool (within the project's root directory): `npm install --save-dev solidity-coverage`
4. Ran the coverage tool from the project's root directory: `./node_modules/.bin/solidity-coverage`
5. Flattened the source code using `truffle-flattener` to accommodate the auditing tools.
6. Installed the Mythril tool from Pypi: `pip3 install mythril`
7. Ran the Mythril tool on each contract: `myth -x path/to/contract`
8. Ran the Securify tool: `java -Xmx6048m -jar securify-0.1.jar -fs contract.sol`
9. Cloned the MAIAN tool: `git clone --depth 1 https://github.com/MAIAN-tool/MAIAN.git maian`
10. Ran the MAIAN tool on each contract: `cd maian/tool/ && python3 maian.py -s path/to/contract contract.sol`
11. Installed the Slither tool: `pip install slither-analyzer`
12. Run Slither from the project directory `slither .`

Assessment

Findings

QSP-1 Funds may be locked if `setRuleAndIntent` is called multiple times

Severity: *High Risk*

Status: Resolved

File(s) affected: `Delegate.sol`, `Indexer.sol`

Description: The function `Delegate.setRuleAndIntent` sets the stake of the delegate owner in the `Indexer` contract by transferring staking tokens from the user to the indexer, through the `Delegate` contract. However, if the function is called a second time, the underlying `Indexer.setIntent` will only transfer a partial amount of the total transferred tokens (i.e., the delta of the previously set intent value versus the currently set intent value). Since the behavior of the `Delegate` and the `Indexer` differ in this regard, tokens can become stuck in the Delegate.
This is elaborated upon in issue 274.

Recommendation: Ensure that the token transfer logic of `Delegate.setRuleAndIntent` and `Indexer.setIntent` are compatible.
Update This issue has been fixed as of pull request 277.

QSP-2 Centralization of Power

Severity: *Medium Risk*

Status: Resolved

File(s) affected: `Indexer.sol`, `Index.sol`

Description: Smart contracts will often have `owner` variables to designate the person with special privileges to make modifications to the smart contract. However, 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.
In particular, the owner may `selfdestruct` the contract locking funds forever. Since the `killContract()` function does not send any of the transferred tokens out of the contract, all tokens will remain permanently locked in the contract if not previously removed by users.
The platform can censor transaction via `unsetIntentForUser()`: whenever an intent is set, the owner can just unset it. It is also not easy to see if it is the owners who did this, as the event emitted is the same.
The owner may also permanently pause the contract locking funds.

Recommendation: Users should be made aware of the roles and responsibilities of Fluidity as a central authority to these contracts. Consider removing the `killContract()` function if it is not necessary. As the `unsetIntentForUser()` function is intended to assist users during the contract being paused, it may be sensible to add the modifier `paused` to ensure it is not doing censorship.

Update: This has been fixed by removing the pausing functionality, `unsetIntentForUser()`, and `killContract()`.

QSP-3 Integer arithmetic may cause incorrect pricing logic

Severity: *Medium Risk*

Status: Resolved

File(s) affected: `Delegate.sol`

Description: On L233 and L290, we have the following two conditions that relate sender and signer order amounts:

- L233 (Equation "A"): `order.sender.param == order.signer.param.mul(10 ** rule.priceExp).div(rule.priceCoef)`
- L290 (Equation "B"): `signerParam = senderParam.mul(rule.priceCoef).div(10 ** rule.priceExp);`

Due to integer arithmetic truncation issues, these two equations may not relate as expected. Consider the case where:

- rule.priceExp = 2
- rule.priceCoef = 3

For Equation B, when senderParam = 90, we obtain `signerParam = 90 * 3 // 100 = 2` due to integer truncation.
However, when plugging back into Equation A, we obtain `2 * 100 // 3 = 66` (which doesn't equal the expected value of 90` .
This means that if the signerParam is calculated by the logic in Equation B, it would not pass the requirement of Equation A.

Recommendation: Consider adding checks to ensure that order amounts behave correctly with respect to these two equations.
Update: This is fixed as of the latest commit. In particular, the case where the signer invokes `_calculateSenderParam()`, and then the values are plugged into `_calculateSignerParam()` should work as intended.

QSP-4 `transferFrom()` success should not be checked by querying token balances

Severity: *Low Risk*

Status: Resolved

File(s) affected: `Swap.sol`

Description: On L349: `require(initialBalance.sub(param) == INRERC20(token).balanceOf(from), "TRANSFER_FAILED");` is a dangerous equality. Although this will hold for most ERC20 tokens, the specification does not guarantee that the external ERC20 contract will adhere to this condition. For example, the token could mint or burn tokens upon a `transfer()` for various reasons.

Recommendation: We recommend removing this balance check require-statement (along with the `initialBalance` assignment on L343), and instead wrap L346 in a require-statement, as discussed in the separate finding "Return values of ERC20 function calls are not checked".

QSP-5 `isValid()` does not check that the `validator` contract is correct

Severity: *Low Risk*

Status: Resolved

File(s) affected: `Swap.sol`, `Types.sol`

Description: While the `Swap` contract ensures that an `Order` is correctly formatted and properly signed in the `isValid()` function, it does not check that the intended `Swap` contract, as denoted by the `Order.signature.validator` field, corresponds to the correct `Swap` contract. If a sender/signer participates with multiple swap contracts, replay attacks may be possible by re-submitting the order.

Recommendation: The function `isValid` should add a check `require(order.signature.validator == address(this))`.
Related to this, in the EIP712-related hash `Types.DOMAIN_TYPEHASH`(L52-57): it may be beneficial to include `chainId` in order to prevent attacks that uses a signature that was signed in the testnet become valid in the mainnet.
Update: Fluidity has clarified to us that the `order.signature.validator` field is only used for informational purposes, and the encoding of the `DOMAIN_SEPARATOR`, which includes the `Swap` address, mitigates this issue.

QSP-6 Unchecked Return Value

Severity: *Low Risk*

Status: Resolved

File(s) affected: `Wrapper.sol`, `Swap.sol`

Description: Most functions will return a `true` or `false` value upon success. Some functions, like `send()`, are more crucial to check than others. It's important to ensure that every necessary function is checked.
On L151 of `Wrapper`, the external `call.value()` is not checked for success, which may cause ether transfers to the user to fail. A similar issue exists for the `transfer()` on L127 of `Wrapper`, and L340 of `Swap`.

Recommendation: The external `call.value()` result should be checked for success by changing the line to: `(bool success, ) = msg.sender.call.value(order.signer.param)("");` followed by a check on `success`.
Additionally, on L127, the return value of `Wrapper.transfer()`should also be checked.
Update: This has been fixed by adding a check on the external call in `Wrapper.sol`. It has also been confirmed that any external calls to the `WETH.sol` contract, the return value does not need to be checked, as the contract reverts on failure instead of returning false.

QSP-7 Gas Usage / `for` Loop Concerns

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `Swap.sol`, `Index.sol`

Description: Gas usage is a main 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 a `for` loop requires too much gas to exit, then it may prevent the contract from functioning correctly entirely. It is best to break such loops into individual functions as possible.
In particular, the `Swap.cancel()` function may fail if the array of `nonces` is too long. Additionally, the `_getEntryLowerThan()` function may need to iterate over many entries, which may make `setLocator()` susceptible to DOS attacks if the array becomes too long.

Recommendation: Although the user could re-invoke the `Swap.cancel()` function by breaking the array up across multiple transactions, we recommend adding a comment to the function's description to indicate such potential issues.
In `Index.setLocator()`, it may be beneficial to have an optional `nextEntry` parameter (which can be computed offline), rather than always invoking `_getEntryLowerThan()` at the cost of extra gas.
Update: A comment has been added to `Swap.cancel()` as suggested. Gas analysis has been performed on `Index.setLocator()` suggesting that gas-related denial-of-service attacks are unlikely, as discussed in Issue 296. Further, if a staker stakes more tokens, they can increase their rank in the Index and reduce their overall gas costs.

QSP-8 Return values of ERC20 function calls are not checked

Severity: *Informational*

Status: Resolved

File(s) affected: `Swap.sol`, `INRERC20.sol`

Description: On L346 of `Swap.sol`, `ERC20.transferFrom()` is expected to return a boolean value indicating success. Although the `INRERC20` is used here, the underlying contract is most likely an `ERC20` token, and therefore its return value should be checked for success.

Recommendation: We recommend removing `INERC20` and instead using `IERC20`. The return value of `transferFrom()` should be checked for success.
Update: This has been fixed through the use of `safeTransferFrom()`.

QSP-9 Unchecked constructor argument

Severity: *Informational*

File(s) affected: `Swap.sol`

Description: In the constructor, `swapRegistry` is passed in but never checked to be non-zero. This may lead to incorrect deployments of `Swap`.

Recommendation: Add a require-statement that checks that `swapRegistry != address(0)`.

Automated Analyses

Maian

Maian did not report any vulnerabilities.

Mythril

Mythril did not report any vulnerabilities.

Securify

Securify reported a few potential "Locked Ether" and "Missing Input Validation" issues, however since the lines associated with these issues were unrelated to the vulnerability types (e.g., comments or contract definitions), they were classified as false positives.

Slither

Slither reported several issues:

1. In `Wrapper.sol`, the return value of several external calls is not checked:

    - L127: `wethContract.transfer()`
    - L144: `wethContract.transferFrom()`
    - L151: `msg.sender.call.value(order.signer.param)("");`

We recommend wrapping the first two calls with `require`, and checking the success of the `call.value()`.

1. In `Swap.sol`, Slither detects that L349: `require(initialBalance.sub(param) == INRERC20(token).balanceOf(from), "TRANSFER_FAILED");` is a dangerous equality, as discussed above. We recommend removing this require-statement.

2. In `INERC20.sol`, Slither warns that the ERC20 specification is not strictly adhered, since the boolean return values of `transfer()` and `transferFrom()` have been removed. We recommend using the IERC20 interface without modification. As a result, we further recommend wrapping the call on L346 of `Swap.sol` with a require-statement.

Update: Fluidity has addressed all concerns related to these findings.

Adherence to Specification

The code adheres to the provided specification.

Code Documentation

The code is well documented and properly commented.

Adherence to Best Practices

The code generally adheres to best practices. We note the following minor issues/questions:

- **Update: confirmed that these are necessary for testing.** It is not clear why the `Imports.sol` files are needed. Can these be removed?

- **Update: fixed through the removal of pausing functionality.** Both `Indexer.sol` and `Wrapper.sol` could inherit from the standard OpenZeppelin `Pausable` smart contract.

- The view function `DelegateFactory.has()` may incorrectly return true if the low-order 20 bits correspond to a deployed address and any of the higher order 12 bits are non-zero. We recommend returning false if any of these higher-order bits are set to one.

- **Update: fixed.** On L52 of `Delegate.sol`, we have that `ERC20_INTERFACE_ID = 0x277f8169`, as computed by the following expression: `bytes4(keccak256('transfer(address,uint256)')) ^ bytes4(keccak256('transferFrom(address,address,uint256)')) ^ bytes4(keccak256('balanceOf(address)')) ^ bytes4(keccak256('allowance(address,address)'))`. However, this computation does not include all functions in the `ERC20` functions, namely `approve(address, uint256)` and `totalSupply()`. It may be better to include these in the hash computation as this would be the more standard `ERC20` interface, and presumably the one that a token would publish to indicate it is ERC20-compliant.

- It is not clear how users or the web interface will utilize `Delegate.getMaxQuote()`, however if the user sets too large of values in `setRule()`, it can cause SafeMath to revert when invoking `getMaxQuote()`. It may be useful to add `require(getMaxQuote(...) > 0)` when invoking `setRule()` in order to ensure that overflow/underflow will not cause SafeMath to revert.

- **Update: confirmed as expected behavior to default to ERC20 unless ERC721 is detected.** In `Swap.transferToken()`, it may be best to check that `kind` is either `ERC721_INTERFACE_ID` or `ERC20_INTERFACE_ID`. With the current setup, the else-branch may accept orders that do not have a correct `kind` value.

- On L52 of `Swap.sol`, it appears that `signerNonceStatus` could simply map to a `bool` instead of a `byte`.

- **Update: fixed.** In `Swap.authorizeSender()` and `Swap.authorizeSigner()` these functions may emit an `AuthorizeSender` or `AuthorizeSigner` event, even if the entries already exist in the mapping. It may be better to first check if the corresponding mapping entries already exist in these functions. Similarly, the `revokeSender()` and `revokeSigner()` functions may emit events, even if the entry did not previously exist in the mapping.

- **Update: fixed.** In `Delegate.sol`, consider adding two helper functions for computing price constraints as opposed to duplication on lines L234, L291, L323, L356.

- **Update: fixed.** In `Delegate.sol`, in the comment on L138, `senderToken` should be `signerToken`.

- **Update: fixed.** The function name `Swap.invalidate()` is not very clear: invalidate(50) seems like it should invalidate the order with nonce 50, but it only invalidates up to 49. Consider alternative names such as "invalidateBefore".

- **Update: confirmed as expected behavior.** It is possible to first `invalidate(50)` then `invalidate(30)` later. This makes it possible to make orders be valid again after they have been canceled in the first place. If this is not an intended behavior, to prevent unexpected consequence, we

- suggest adding a check that the `minimumNonce` be larger than `signerMinimumNonce[msg.sender]`.

Test Results

Test Suite Results

```
yarn test
yarn run v1.19.2
$ yarn clean && yarn compile && lerna run test --concurrency=1
$ lerna run clean
lerna notice cli v3.19.0
lerna info versioning independent
lerna info Executing command in 9 packages: "yarn run clean"
lerna info run Ran npm script 'clean' in '@airswap/tokens' in 0.3s:
$ rm -rf ./build
lerna info run Ran npm script 'clean' in '@airswap/transfers' in 0.3s:
$ rm -rf ./build
lerna info run Ran npm script 'clean' in '@airswap/types' in 0.2s:
$ rm -rf ./build
lerna info run Ran npm script 'clean' in '@airswap/indexer' in 0.3s:
$ rm -rf ./build
lerna info run Ran npm script 'clean' in '@airswap/swap' in 0.3s:
$ rm -rf ./build
lerna info run Ran npm script 'clean' in '@airswap/deployer' in 0.3s:
$ rm -rf ./build &&  rm -rf ./flatten
lerna info run Ran npm script 'clean' in '@airswap/delegate' in 0.3s:
$ rm -rf ./build
lerna info run Ran npm script 'clean' in '@airswap/pre-swap-checker' in 0.3s:
$ rm -rf ./build
lerna info run Ran npm script 'clean' in '@airswap/wrapper' in 0.3s:
$ rm -rf ./build
lerna success run Ran npm script 'clean' in 9 packages in 1.3s:
lerna success - @airswap/delegate
lerna success - @airswap/indexer
lerna success - @airswap/swap
lerna success - @airswap/tokens
lerna success - @airswap/transfers
lerna success - @airswap/types
lerna success - @airswap/wrapper
lerna success - @airswap/deployer
lerna success - @airswap/pre-swap-checker
$ lerna run compile
lerna notice cli v3.19.0
lerna info versioning independent
lerna info Executing command in 9 packages: "yarn run compile"
lerna info run Ran npm script 'compile' in '@airswap/tokens' in 10.6s:
$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/AdaptedERC721.sol
> Compiling ./contracts/AdaptedKittyERC721.sol
> Compiling ./contracts/ERC1155.sol
> Compiling ./contracts/FungibleToken.sol
> Compiling ./contracts/IERC721Receiver.sol
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/MintableERC1155Token.sol
> Compiling ./contracts/NonFungibleToken.sol
> Compiling ./contracts/OMGToken.sol
> Compiling ./contracts/OrderTest721.sol
> Compiling ./contracts/WETH9.sol
> Compiling ./contracts/interfaces/IERC1155.sol
> Compiling ./contracts/interfaces/IERC1155Receiver.sol
> Compiling ./contracts/interfaces/IWETH.sol
> Compiling openzeppelin-solidity/contracts/GSN/Context.sol
> Compiling openzeppelin-solidity/contracts/access/Roles.sol
> Compiling openzeppelin-solidity/contracts/access/roles/MinterRole.sol
> Compiling openzeppelin-solidity/contracts/drafts/Counters.sol
> Compiling openzeppelin-solidity/contracts/introspection/ERC165.sol
> Compiling openzeppelin-solidity/contracts/introspection/IERC165.sol
> Compiling openzeppelin-solidity/contracts/math/SafeMath.sol
> Compiling openzeppelin-solidity/contracts/ownership/Ownable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC721/IERC721Receiver.sol
> Compiling openzeppelin-solidity/contracts/utils/Address.sol
> Artifacts written to /Users/ezulkosk/audits/airswap-protocols/source/tokens/build/contracts
> Compiled successfully using:
   - solc: 0.5.12+commit.7709ece9.Emscripten.clang

lerna info run Ran npm script 'compile' in '@airswap/types' in 10.8s:
$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/Imports.sol
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/Types.sol
> Compiling @airswap/tokens/contracts/AdaptedERC721.sol
> Compiling @airswap/tokens/contracts/FungibleToken.sol
> Compiling @airswap/tokens/contracts/NonFungibleToken.sol
> Compiling @gnosis.pm/mock-contract/contracts/MockContract.sol
> Compiling openzeppelin-solidity/contracts/GSN/Context.sol
> Compiling openzeppelin-solidity/contracts/access/Roles.sol
> Compiling openzeppelin-solidity/contracts/access/roles/MinterRole.sol
> Compiling openzeppelin-solidity/contracts/drafts/Counters.sol
> Compiling openzeppelin-solidity/contracts/introspection/ERC165.sol
> Compiling openzeppelin-solidity/contracts/introspection/IERC165.sol
> Compiling openzeppelin-solidity/contracts/math/SafeMath.sol
```

```
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC721/IERC721Receiver.sol
> Compiling openzeppelin-solidity/contracts/utils/Address.sol

    > compilation warnings encountered:

/Users/ezulkosk/audits/airswap-protocols/source/types/contracts/Types.sol:18:1: Warning: Experimental features
are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^

> Artifacts written to /Users/ezulkosk/audits/airswap-protocols/source/types/build/contracts
> Compiled successfully using:
   - solc: 0.5.12+commit.7709ece9.Emscripten.clang

lerna info run Ran npm script 'compile' in '@airswap/transfers' in 12.4s:
$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/Imports.sol
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/TransferHandlerRegistry.sol
> Compiling ./contracts/handlers/ERC1155TransferHandler.sol
> Compiling ./contracts/handlers/ERC20TransferHandler.sol
> Compiling ./contracts/handlers/ERC721TransferHandler.sol
> Compiling ./contracts/handlers/KittyCoreTransferHandler.sol
> Compiling ./contracts/interfaces/IKittyCoreTokenTransfer.sol
> Compiling ./contracts/interfaces/ITransferHandler.sol
> Compiling @airswap/tokens/contracts/AdaptedERC721.sol
> Compiling @airswap/tokens/contracts/AdaptedKittyERC721.sol
> Compiling @airswap/tokens/contracts/ERC1155.sol
> Compiling @airswap/tokens/contracts/FungibleToken.sol
> Compiling @airswap/tokens/contracts/MintableERC1155Token.sol
> Compiling @airswap/tokens/contracts/NonFungibleToken.sol
> Compiling @airswap/tokens/contracts/interfaces/IERC1155.sol
> Compiling @airswap/tokens/contracts/interfaces/IERC1155Receiver.sol
> Compiling @gnosis.pm/mock-contract/contracts/MockContract.sol
> Compiling openzeppelin-solidity/contracts/GSN/Context.sol
> Compiling openzeppelin-solidity/contracts/access/Roles.sol
> Compiling openzeppelin-solidity/contracts/access/roles/MinterRole.sol
> Compiling openzeppelin-solidity/contracts/drafts/Counters.sol
> Compiling openzeppelin-solidity/contracts/introspection/ERC165.sol
> Compiling openzeppelin-solidity/contracts/introspection/IERC165.sol
> Compiling openzeppelin-solidity/contracts/math/SafeMath.sol
> Compiling openzeppelin-solidity/contracts/ownership/Ownable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/SafeERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC721/IERC721.sol
> Compiling openzeppelin-solidity/contracts/token/ERC721/IERC721Receiver.sol
> Compiling openzeppelin-solidity/contracts/utils/Address.sol
> Artifacts written to /Users/ezulkosk/audits/airswap-protocols/source/transfers/build/contracts
> Compiled successfully using:
   - solc: 0.5.12+commit.7709ece9.Emscripten.clang

lerna info run Ran npm script 'compile' in '@airswap/deployer' in 4.3s:
$ truffle compile

Compiling your contracts...
===========================
> Everything is up to date, there is nothing to compile.

lerna info run Ran npm script 'compile' in '@airswap/indexer' in 13.6s:
$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/Imports.sol
> Compiling ./contracts/Index.sol
> Compiling ./contracts/Indexer.sol
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/interfaces/IIndexer.sol
> Compiling ./contracts/interfaces/ILocatorWhitelist.sol
> Compiling @airswap/delegate/contracts/Delegate.sol
> Compiling @airswap/delegate/contracts/DelegateFactory.sol
> Compiling @airswap/delegate/contracts/interfaces/IDelegate.sol
> Compiling @airswap/delegate/contracts/interfaces/IDelegateFactory.sol
> Compiling @airswap/indexer/contracts/interfaces/IIndexer.sol
> Compiling @airswap/indexer/contracts/interfaces/ILocatorWhitelist.sol
> Compiling @airswap/swap/contracts/Swap.sol
> Compiling @airswap/swap/contracts/interfaces/ISwap.sol
> Compiling @airswap/tokens/contracts/FungibleToken.sol
> Compiling @airswap/transfers/contracts/TransferHandlerRegistry.sol
> Compiling @airswap/transfers/contracts/handlers/ERC20TransferHandler.sol
> Compiling @airswap/transfers/contracts/interfaces/ITransferHandler.sol
> Compiling @airswap/types/contracts/Types.sol
> Compiling @gnosis.pm/mock-contract/contracts/MockContract.sol
> Compiling openzeppelin-solidity/contracts/GSN/Context.sol
> Compiling openzeppelin-solidity/contracts/access/Roles.sol
> Compiling openzeppelin-solidity/contracts/access/roles/MinterRole.sol
> Compiling openzeppelin-solidity/contracts/math/SafeMath.sol
> Compiling openzeppelin-solidity/contracts/ownership/Ownable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/SafeERC20.sol
> Compiling openzeppelin-solidity/contracts/utils/Address.sol

    > compilation warnings encountered:
```

```
@airswap/types/contracts/Types.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,@airswap/delegate/contracts/interfaces/IDelegate.sol:18:1: Warning: Experimental features are turned on. Do not
use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,@airswap/swap/contracts/interfaces/ISwap.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,@airswap/delegate/contracts/Delegate.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,@airswap/swap/contracts/Swap.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/indexer/contracts/Index.sol:17:1: Warning: Experimental
features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/indexer/contracts/Indexer.sol:18:1: Warning: Experimental
features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^

> Artifacts written to /Users/ezulkosk/audits/airswap-protocols/source/indexer/build/contracts
> Compiled successfully using:
   - solc: 0.5.12+commit.7709ece9.Emscripten.clang

lerna info run Ran npm script 'compile' in '@airswap/swap' in 14.1s:
$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/Imports.sol
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/Swap.sol
> Compiling ./contracts/interfaces/ISwap.sol
> Compiling @airswap/swap/contracts/interfaces/ISwap.sol
> Compiling @airswap/tokens/contracts/AdaptedERC721.sol
> Compiling @airswap/tokens/contracts/AdaptedKittyERC721.sol
> Compiling @airswap/tokens/contracts/ERC1155.sol
> Compiling @airswap/tokens/contracts/FungibleToken.sol
> Compiling @airswap/tokens/contracts/MintableERC1155Token.sol
> Compiling @airswap/tokens/contracts/NonFungibleToken.sol
> Compiling @airswap/tokens/contracts/OMGToken.sol
> Compiling @airswap/tokens/contracts/interfaces/IERC1155.sol
> Compiling @airswap/tokens/contracts/interfaces/IERC1155Receiver.sol
> Compiling @airswap/transfers/contracts/TransferHandlerRegistry.sol
> Compiling @airswap/transfers/contracts/handlers/ERC1155TransferHandler.sol
> Compiling @airswap/transfers/contracts/handlers/ERC20TransferHandler.sol
> Compiling @airswap/transfers/contracts/handlers/ERC721TransferHandler.sol
> Compiling @airswap/transfers/contracts/handlers/KittyCoreTransferHandler.sol
> Compiling @airswap/transfers/contracts/interfaces/IKittyCoreTokenTransfer.sol
> Compiling @airswap/transfers/contracts/interfaces/ITransferHandler.sol
> Compiling @airswap/types/contracts/Types.sol
> Compiling @gnosis.pm/mock-contract/contracts/MockContract.sol
> Compiling openzeppelin-solidity/contracts/GSN/Context.sol
> Compiling openzeppelin-solidity/contracts/access/Roles.sol
> Compiling openzeppelin-solidity/contracts/access/roles/MinterRole.sol
> Compiling openzeppelin-solidity/contracts/drafts/Counters.sol
> Compiling openzeppelin-solidity/contracts/introspection/ERC165.sol
> Compiling openzeppelin-solidity/contracts/introspection/IERC165.sol
> Compiling openzeppelin-solidity/contracts/math/SafeMath.sol
> Compiling openzeppelin-solidity/contracts/ownership/Ownable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/SafeERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC721/IERC721.sol
> Compiling openzeppelin-solidity/contracts/token/ERC721/IERC721Receiver.sol
> Compiling openzeppelin-solidity/contracts/utils/Address.sol
    > compilation warnings encountered:

@airswap/types/contracts/Types.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,@airswap/swap/contracts/interfaces/ISwap.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/swap/contracts/Swap.sol:18:1: Warning: Experimental features
are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/swap/contracts/interfaces/ISwap.sol:18:1: Warning: Experimental
features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-----------------------------^

> Artifacts written to /Users/ezulkosk/audits/airswap-protocols/source/swap/build/contracts
> Compiled successfully using:
   - solc: 0.5.12+commit.7709ece9.Emscripten.clang

lerna info run Ran npm script 'compile' in '@airswap/pre-swap-checker' in 11.7s:
$ truffle compile
```

```
Compiling your contracts...
===========================
> Compiling ./contracts/Imports.sol
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/PreSwapChecker.sol
> Compiling @airswap/swap/contracts/Swap.sol
> Compiling @airswap/swap/contracts/interfaces/ISwap.sol
> Compiling @airswap/tokens/contracts/AdaptedERC721.sol
> Compiling @airswap/tokens/contracts/FungibleToken.sol
> Compiling @airswap/tokens/contracts/NonFungibleToken.sol
> Compiling @airswap/tokens/contracts/OMGToken.sol
> Compiling @airswap/transfers/contracts/TransferHandlerRegistry.sol
> Compiling @airswap/transfers/contracts/handlers/ERC20TransferHandler.sol
> Compiling @airswap/transfers/contracts/interfaces/ITransferHandler.sol
> Compiling @airswap/types/contracts/Types.sol
> Compiling openzeppelin-solidity/contracts/GSN/Context.sol
> Compiling openzeppelin-solidity/contracts/access/Roles.sol
> Compiling openzeppelin-solidity/contracts/access/roles/MinterRole.sol
> Compiling openzeppelin-solidity/contracts/drafts/Counters.sol
> Compiling openzeppelin-solidity/contracts/introspection/ERC165.sol
> Compiling openzeppelin-solidity/contracts/introspection/IERC165.sol
> Compiling openzeppelin-solidity/contracts/math/SafeMath.sol
> Compiling openzeppelin-solidity/contracts/ownership/Ownable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/SafeERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC721/IERC721.sol
> Compiling openzeppelin-solidity/contracts/token/ERC721/IERC721Receiver.sol
> Compiling openzeppelin-solidity/contracts/utils/Address.sol

    > compilation warnings encountered:

@airswap/types/contracts/Types.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
,@airswap/swap/contracts/interfaces/ISwap.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
,@airswap/swap/contracts/Swap.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
,/Users/ezulkosk/audits/airswap-protocols/utils/pre-swap-checker/contracts/PreSwapChecker.sol:2:1: Warning:
Experimental features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
> Artifacts written to /Users/ezulkosk/audits/airswap-protocols/utils/pre-swap-checker/build/contracts
> Compiled successfully using:
   - solc: 0.5.12+commit.7709ece9.Emscripten.clang

lerna info run Ran npm script 'compile' in '@airswap/delegate' in 13.0s:
$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/Delegate.sol
> Compiling ./contracts/DelegateFactory.sol
> Compiling ./contracts/Imports.sol
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/interfaces/IDelegate.sol
> Compiling ./contracts/interfaces/IDelegateFactory.sol
> Compiling @airswap/delegate/contracts/interfaces/IDelegate.sol
> Compiling @airswap/indexer/contracts/Index.sol
> Compiling @airswap/indexer/contracts/Indexer.sol
> Compiling @airswap/indexer/contracts/interfaces/IIndexer.sol
> Compiling @airswap/indexer/contracts/interfaces/ILocatorWhitelist.sol
> Compiling @airswap/swap/contracts/Swap.sol
> Compiling @airswap/swap/contracts/interfaces/ISwap.sol
> Compiling @airswap/tokens/contracts/FungibleToken.sol
> Compiling @airswap/transfers/contracts/TransferHandlerRegistry.sol
> Compiling @airswap/transfers/contracts/handlers/ERC20TransferHandler.sol
> Compiling @airswap/transfers/contracts/interfaces/ITransferHandler.sol
> Compiling @airswap/types/contracts/Types.sol
> Compiling @gnosis.pm/mock-contract/contracts/MockContract.sol
> Compiling openzeppelin-solidity/contracts/GSN/Context.sol
> Compiling openzeppelin-solidity/contracts/access/Roles.sol
> Compiling openzeppelin-solidity/contracts/access/roles/MinterRole.sol
> Compiling openzeppelin-solidity/contracts/math/SafeMath.sol
> Compiling openzeppelin-solidity/contracts/ownership/Ownable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/SafeERC20.sol
> Compiling openzeppelin-solidity/contracts/utils/Address.sol

    > compilation warnings encountered:

@airswap/types/contracts/Types.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
,@airswap/delegate/contracts/interfaces/IDelegate.sol:18:1: Warning: Experimental features are turned on. Do not
use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
,@airswap/swap/contracts/interfaces/ISwap.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
```

```
,/Users/ezulkosk/audits/airswap-protocols/source/delegate/contracts/Delegate.sol:18:1: Warning: Experimental
features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/swap/contracts/Swap.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/indexer/contracts/Index.sol:17:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/indexer/contracts/Indexer.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/delegate/contracts/interfaces/IDelegate.sol:18:1: Warning:
Experimental features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^

> Artifacts written to /Users/ezulkosk/audits/airswap-protocols/source/delegate/build/contracts
> Compiled successfully using:
   - solc: 0.5.12+commit.7709ece9.Emscripten.clang

lerna info run Ran npm script 'compile' in '@airswap/wrapper' in 12.4s:
$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/HelperMock.sol
> Compiling ./contracts/Imports.sol
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/Wrapper.sol
> Compiling @airswap/delegate/contracts/Delegate.sol
> Compiling @airswap/delegate/contracts/interfaces/IDelegate.sol
> Compiling @airswap/indexer/contracts/Index.sol
> Compiling @airswap/indexer/contracts/Indexer.sol
> Compiling @airswap/indexer/contracts/interfaces/IIndexer.sol
> Compiling @airswap/indexer/contracts/interfaces/ILocatorWhitelist.sol
> Compiling @airswap/swap/contracts/Swap.sol
> Compiling @airswap/swap/contracts/interfaces/ISwap.sol
> Compiling @airswap/tokens/contracts/FungibleToken.sol
> Compiling @airswap/tokens/contracts/WETH9.sol
> Compiling @airswap/tokens/contracts/interfaces/IWETH.sol
> Compiling @airswap/transfers/contracts/TransferHandlerRegistry.sol
> Compiling @airswap/transfers/contracts/handlers/ERC20TransferHandler.sol
> Compiling @airswap/transfers/contracts/interfaces/ITransferHandler.sol
> Compiling @airswap/types/contracts/Types.sol
> Compiling @gnosis.pm/mock-contract/contracts/MockContract.sol
> Compiling openzeppelin-solidity/contracts/GSN/Context.sol
> Compiling openzeppelin-solidity/contracts/access/Roles.sol
> Compiling openzeppelin-solidity/contracts/access/roles/MinterRole.sol
> Compiling openzeppelin-solidity/contracts/math/SafeMath.sol
> Compiling openzeppelin-solidity/contracts/ownership/Ownable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
> Compiling openzeppelin-solidity/contracts/token/ERC20/SafeERC20.sol
> Compiling openzeppelin-solidity/contracts/utils/Address.sol

    > compilation warnings encountered:

@airswap/types/contracts/Types.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/swap/contracts/interfaces/ISwap.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/delegate/contracts/interfaces/IDelegate.sol:18:1: Warning: Experimental features are turned on. Do not
use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/wrapper/contracts/Wrapper.sol:18:1: Warning: Experimental
features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/wrapper/contracts/HelperMock.sol:2:1: Warning: Experimental
features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/swap/contracts/Swap.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/delegate/contracts/Delegate.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/indexer/contracts/Index.sol:17:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^
,@airswap/indexer/contracts/Indexer.sol:18:1: Warning: Experimental features are turned on. Do not use
experimental features on live deployments.
pragma experimental ABIEncoderV2;
^------------------------------^

> Artifacts written to /Users/ezulkosk/audits/airswap-protocols/source/wrapper/build/contracts
> Compiled successfully using:
   - solc: 0.5.12+commit.7709ece9.Emscripten.clang
```

```
lerna success run Ran npm script 'compile' in 9 packages in 62.4s:
lerna success - @airswap/delegate
lerna success - @airswap/indexer
lerna success - @airswap/swap
lerna success - @airswap/tokens
lerna success - @airswap/transfers
lerna success - @airswap/types
lerna success - @airswap/wrapper
lerna success - @airswap/deployer
lerna success - @airswap/pre-swap-checker
lerna notice cli v3.19.0
lerna info versioning independent
lerna info Executing command in 9 packages: "yarn run test"
lerna info run Ran npm script 'test' in '@airswap/order-utils' in 1.4s:
$ mocha test


  Orders
     ✓ Checks that a generated order is valid

  Signatures
     ✓ Checks that a Version 0x45: personalSign signature is valid
     ✓ Checks that a Version 0x01: signTypedData signature is valid


  3 passing (27ms)

lerna info run Ran npm script 'test' in '@airswap/transfers' in 10.1s:
$ truffle test
Using network 'development'.


Compiling your contracts...
===========================
> Everything is up to date, there is nothing to compile.



  Contract: TransferHandlerRegistry Unit Tests
    Test fetching non-existent handler
       ✓ test fetching non-existent handler, returns null address
    Test adding handler to registry
       ✓ test adding, should pass (87ms)
    Test adding an existing handler from registry will fail
       ✓ test adding an existing handler will fail (119ms)

  Contract: TransferHandlerRegistry
    Deploying...
       ✓ Deployed TransferHandlerRegistry contract (56ms)
       ✓ Deployed test contract "AST" (55ms)
       ✓ Deployed test contract "MintableERC1155Token" (71ms)
       ✓ Test adding transferHandler by non-owner reverts (46ms)
       ✓ Set up TokenRegistry (561ms)
    Minting ERC20 tokens (AST)...
       ✓ Mints 1000 AST for Alice (67ms)
    Approving ERC20 tokens (AST)...
       ✓ Checks approvals (Alice 250 AST (62ms)
    ERC20 TransferHandler
       ✓ Checks balances and allowances for Alice and Bob... (99ms)
       ✓ Checks that Alice cannot trade 200 AST more than allowance of 50 AST (136ms)
       ✓ Checks remaining balances and approvals were not updated in failed transfer (86ms)
       ✓ Adding an id with ERC20TransferHandler will cause revert (51ms)
    ERC721 and CKitty TransferHandler
       ✓ Deployed test ERC721 contract "ConcertTicket" (70ms)
       ✓ Deployed test contract "CKITTY" (51ms)
       ✓ Carol initiates an ERC721 transfer of ConcertTicket #121 tokens from Alice to Bob (224ms)
       ✓ Carol fails to perform transfer of ConcertTicket #121 from Bob to Alice when an amount is set (103ms)
       ✓ Mints a kitty collectible (#54321) for Bob (78ms)
       ✓ Bob approves KittyCoreHandler to transfer his kitty collectible (47ms)
       ✓ Carol fails to perform transfer of CKITTY collectable #54321 from Bob to Alice when an amount is set
(79ms)
       ✓ Carol initiates a transfer of CKITTY collectable #54321 from Bob to Alice (72ms)
    ERC1155 TransferHandler
       ✓ Mints 100 of Dragon game token (#10) for Alice (65ms)
       ✓ Check the Dragon game token (#10) balance prior to transfer (39ms)
       ✓ Alice approves ERC115TransferHandler to transfer all the her ERC1155 tokens (38ms)
       ✓ Carol initiates an ERC1155 transfer of Dragon game token (#10) from Alice to Bob (107ms)


  26 passing (4s)

lerna info run Ran npm script 'test' in '@airswap/types' in 9.2s:
$ truffle test
Using network 'development'.


Compiling your contracts...
===========================
> Compiling ./test/MockTypes.sol

    > compilation warnings encountered:

/Users/ezulkosk/audits/airswap-protocols/source/types/contracts/Types.sol:18:1: Warning: Experimental features
are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/types/test/MockTypes.sol:2:1: Warning: Experimental features
are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
```

```
  Contract: Types Unit Tests
    Test hashing functions within the library
       ✓ Test hashOrder (163ms)
       ✓ Test hashDomain (56ms)


  2 passing (2s)

lerna info run Ran npm script 'test' in '@airswap/indexer' in 26.4s:
$ truffle test
Using network 'development'.


Compiling your contracts...
===========================
> Compiling ./contracts/interfaces/IIndexer.sol
> Compiling ./contracts/interfaces/ILocatorWhitelist.sol



  Contract: Index Unit Tests
    Test constructor
       ✓ should setup the linked locators as just a head, length 0 (48ms)
    Test setLocator
       ✓ should not allow a non owner to call setLocator (91ms)
       ✓ should not allow a blank locator to be set (49ms)
       ✓ should allow an entry to be inserted by the owner (112ms)
       ✓ should insert subsequent entries in the correct order (230ms)
       ✓ should insert an identical stake after the pre-existing one (281ms)
       ✓ should not be able to set a second locator if one already exists for an address (115ms)
    Test updateLocator
       ✓ should not allow a non owner to call updateLocator (49ms)
       ✓ should not allow update to non-existent locator (51ms)
       ✓ should not allow update to a blank locator (121ms)
       ✓ should allow an entry to be updated by the owner (161ms)
       ✓ should update the list order on updated score (287ms)
    Test getting entries
       ✓ should return the entry of a user (73ms)
       ✓ should return empty entry for an unset user
    Test unsetLocator
       ✓ should not allow a non owner to call unsetLocator (43ms)
       ✓ should leave state unchanged for someone who hasnt staked (100ms)
       ✓ should unset the entry for a valid user (230ms)
    Test getScore
       ✓ should return no score for a non-user (101ms)
       ✓ should return the correct score for a valid user
    Test getLocator
       ✓ should return empty locator for a non-user (78ms)
       ✓ should return the correct locator for a valid user (38ms)
    Test getLocators
       ✓ returns an array of empty locators
       ✓ returns specified number of elements if < length (267ms)
       ✓ returns only length if requested number if larger (198ms)
       ✓ starts the array at the specified starting user (190ms)
       ✓ starts the array at the specified starting user - longer list (543ms)
       ✓ returns nothing for an unstaked user (205ms)

  Contract: Indexer Unit Tests
    Check constructor
       ✓ should set the staking token address correctly
    Test createIndex
       ✓ createIndex should emit an event and create a new index (51ms)
       ✓ createIndex should create index for same token pair but different protocol (101ms)
       ✓ createIndex should just return an address if the index exists (88ms)
    Test addTokenToBlacklist and removeTokenFromBlacklist
       ✓ should not allow a non-owner to blacklist a token (47ms)
       ✓ should allow the owner to blacklist a token (56ms)
       ✓ should not emit an event if token is already blacklisted (73ms)
       ✓ should not allow a non-owner to un-blacklist a token (40ms)
       ✓ should allow the owner to un-blacklist a token (128ms)
    Test setIntent
       ✓ should not set an intent if the index doesnt exist (72ms)
       ✓ should not set an intent if the locator is not whitelisted (185ms)
       ✓ should not set an intent if a token is blacklisted (323ms)
       ✓ should not set an intent if the staking tokens arent approved (266ms)
       ✓ should set a valid intent on a non-whitelisted indexer (165ms)
       ✓ should set 2 intents for different protocols on the same market (325ms)
       ✓ should set a valid intent on a whitelisted indexer (230ms)
       ✓ should update an intent if the user has already staked - increase stake (369ms)
       ✓ should fail updating the intent when transfer of staking tokens fails (713ms)
       ✓ should update an intent if the user has already staked - decrease stake (397ms)
       ✓ should update an intent if the user has already staked - same stake (371ms)
    Test unsetIntent
       ✓ should not unset an intent if the index doesnt exist (44ms)
       ✓ should not unset an intent if the intent does not exist (146ms)
       ✓ should successfully unset an intent (294ms)
       ✓ should revert if unset an intent failed in token transfer (387ms)
    Test getLocators
       ✓ should return blank results if the index doesnt exist
       ✓ should return blank results if a token is blacklisted (204ms)
       ✓ should otherwise return the intents (758ms)
    Test getStakedAmount.call
       ✓ should return 0 if the index does not exist
       ✓ should retrieve the score on a token pair for a user (208ms)

  Contract: Indexer
    Deploying...
       ✓ Deployed staking token "AST" (39ms)
       ✓ Deployed trading token "DAI" (43ms)
       ✓ Deployed trading token "WETH" (45ms)
       ✓ Deployed Indexer contract (52ms)
    Index setup
       ✓ Bob creates a index (collection of intents) for WETH/DAI, LIBP2P (68ms)
```

```
    ✓ Bob tries to create a duplicate index (collection of intents) for WETH/DAI, LIBP2P (54ms)
    ✓ Bob tries to create another index for WETH/DAI, but for Delegates (58ms)
    ✓ The owner can set and unset a locator whitelist for a locator type (397ms)
    ✓ Bob ensures no intents are on the Indexer for existing index (54ms)
    ✓ Bob ensures no intents are on the Indexer for non-existing index
    ✓ Alice attempts to stake and set an intent but fails due to no index (53ms)
  Staking
    ✓ Alice attempts to stake with 0 and set an intent succeeds (76ms)
    ✓ Alice attempts to unset an intent and succeeds (64ms)
    ✓ Fails due to no staking token balance (95ms)
    ✓ Staking tokens are minted for Alice and Bob (71ms)
    ✓ Fails due to no staking token allowance (102ms)
    ✓ Alice and Bob approve Indexer to spend staking tokens (64ms)
    ✓ Checks balances
    ✓ Alice attempts to stake and set an intent succeeds (86ms)
    ✓ Checks balances
    ✓ The Alice can unset alice's intent (113ms)
    ✓ Bob can set an intent on 2 indexes for the same market (397ms)
    ✓ Bob can increase his intent stake (193ms)
    ✓ Bob can decrease his intent stake and change his locator (225ms)
    ✓ Bob can keep the same stake amount (158ms)
    ✓ Owner sets the locator whitelist for delegates, and alice cannot set intent (98ms)
    ✓ Deploy a whitelisted delegate for alice (177ms)
    ✓ Bob can remove his unwhitelisted intent from delegate index (89ms)
    ✓ Remove locator whitelist from delegate index (73ms)
  Intent integrity
    ✓ Bob ensures only one intent is on the Index for libp2p (67ms)
    ✓ Alice attempts to unset non-existent index and reverts (50ms)
    ✓ Bob attempts to unset an intent and succeeds (81ms)
    ✓ Alice unsets her intent on delegate index and succeeds (77ms)
    ✓ Bob attempts to unset the intent he just unset and reverts (81ms)
    ✓ Checks balances (46ms)
    ✓ Bob ensures there are no more intents the Index for libp2p (55ms)
    ✓ Alice attempts to set an intent for libp2p and succeeds (91ms)
  Blacklisting
    ✓ Alice attempts to blacklist a index and fails because she is not owner (46ms)
    ✓ Owner attempts to blacklist a index and succeeds (57ms)
    ✓ Bob tries to fetch intent on blacklisted token
    ✓ Owner attempts to blacklist same asset which does not emit a new event (42ms)
    ✓ Alice attempts to stake and set an intent and fails due to blacklist (66ms)
    ✓ Alice attempts to unset an intent and succeeds regardless of blacklist (90ms)
    ✓ Alice attempts to remove from blacklist fails because she is not owner (44ms)
    ✓ Owner attempts to remove non-existent token from blacklist with no event emitted
    ✓ Owner attempts to remove token from blacklist and succeeds (41ms)
    ✓ Alice and Bob attempt to stake and set an intent and succeed (262ms)
    ✓ Bob fetches intents starting at bobAddress (72ms)
    ✓ shouldn't allow a locator of 0 (269ms)
    ✓ shouldn't allow a previous stake to be updated with locator 0 (308ms)


  106 passing (19s)

lerna info run Ran npm script 'test' in '@airswap/swap' in 32.4s:
$ truffle test
Using network 'development'.


Compiling your contracts...
===========================
> Compiling ./contracts/interfaces/ISwap.sol

    > compilation warnings encountered:

@airswap/types/contracts/Types.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/swap/contracts/interfaces/ISwap.sol:18:1: Warning: Experimental
features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^



  Contract: Swap Handler Checks
    Deploying...
    ✓ Deployed Swap contract (1221ms)
    ✓ Deployed test contract "AST" (41ms)
    ✓ Deployed test contract "DAI" (41ms)
    ✓ Deployed test contract "OMG" (52ms)
    ✓ Deployed test contract "MintableERC1155Token" (48ms)
    ✓ Test adding transferHandler by non-owner reverts
    ✓ Set up TokenRegistry (290ms)
  Minting ERC20 tokens (AST, DAI, and OMG)...
    ✓ Mints 1000 AST for Alice (84ms)
    ✓ Mints 1000 OMG for Alice (83ms)
    ✓ Mints 1000 DAI for Bob (69ms)
  Approving ERC20 tokens (AST and DAI)...
    ✓ Checks approvals (Alice 250 AST, 200 OMG, and 0 DAI, Bob 0 AST and 1000 DAI) (238ms)
  Swaps (Fungible)
    ✓ Checks that Bob can swap with Alice (200 AST for 50 DAI) (300ms)
    ✓ Checks balances and allowances for Alice and Bob... (142ms)
    ✓ Checks that Alice cannot trade 200 AST more than allowance of 50 AST (66ms)
    ✓ Checks that Bob can not trade more than 950 DAI balance he holds (513ms)
    ✓ Checks remaining balances and approvals were not updated in failed trades (138ms)
    ✓ Adding an id with Fungible token will cause revert (513ms)
    ✓ Checks that adding an affiliate address still swaps (350ms)
    ✓ Transfers tokens back for future tests (339ms)
  Swaps (Non-standard Fungible)
    ✓ Checks that Bob can swap with Alice (200 OMG for 50 DAI) (299ms)
    ✓ Checks balances... (60ms)
    ✓ Checks that Bob cannot take the same order again (200 OMG for 50 DAI) (41ms)
```

```
    ✓ Checks balances and approvals... (94ms)
    ✓ Checks that Alice cannot trade 200 OMG when allowance is 0 (126ms)
    ✓ Checks that Bob can not trade more OMG tokens than he holds (111ms)
    ✓ Checks remaining balances and approvals (135ms)
Deploying NFT tokens...
    ✓ Deployed test contract "ConcertTicket" (50ms)
    ✓ Deployed test contract "Collectible" (42ms)
Swaps with Fees
    ✓ Checks that Carol gets paid 50 AST for facilitating a trade between Alice and Bob (393ms)
    ✓ Checks balances... (93ms)
    ✓ Checks that Carol gets paid token ticket (#121) for facilitating a trade between Alice and Bob (540ms)
Minting ERC721 Tokens
    ✓ Mints a concert ticket (#12345) for Alice (55ms)
    ✓ Mints a kitty collectible (#54321) for Bob (48ms)
Swaps (Non-Fungible) with unknown kind
    ✓ Alice approves Swap to transfer her concert ticket (38ms)
    ✓ Alice sends Bob with an unknown kind for 100 DAI (492ms)
Swaps (Non-Fungible)
    ✓ Alice approves Swap to transfer her concert ticket
    ✓ Bob cannot buy Ticket #12345 from Alice if she sends id and amount in Party struct (662ms)
    ✓ Bob buys Ticket #12345 from Alice for 100 DAI (303ms)
    ✓ Bob approves Swap to transfer his kitty collectible (40ms)
    ✓ Alice cannot buy Kitty #54321 from Bob for 50 AST if Kitty amount specified (565ms)
    ✓ Alice buys Kitty #54321 from Bob for 50 AST (423ms)
    ✓ Alice approves Swap to transfer her kitty collectible (53ms)
    ✓ Checks that Carol gets paid Kitty #54321 for facilitating a trade between Alice and Bob (389ms)
  Minting ERC1155 Tokens
    ✓ Mints 100 of Dragon game token (#10) for Alice (60ms)
    ✓ Mints 100 of Dragon game token (#15) for Bob (52ms)
  Swaps (ERC-1155)
    ✓ Alice approves Swap to transfer all the her ERC1155 tokens
    ✓ Bob cannot sell 5 Dragon Token (#15) to Alice when he did not approve them (398ms)
    ✓ Check the balances prior to ERC1155 token transfers (170ms)
    ✓ Bob buys 50 Dragon Token (#10) from Alice when she sends id and amount in Party struct (303ms)
    ✓ Checks that Carol gets paid 10 Dragon Token (#10) for facilitating a fungible token transfer trade
between Alice and Bob (409ms)
    ✓ Check the balances after ERC1155 token transfers (161ms)

Contract: Swap Unit Tests
  Test swap
    ✓ test when order is expired (46ms)
    ✓ test when order nonce is too low (86ms)
    ✓ test when sender is provided, and the sender is unauthorized (63ms)
    ✓ test when sender is provided, the sender is authorized, the signature.v is 0, and the signer wallet is
unauthorized (80ms)
    ✓ test swap when sender and signer are the same (87ms)
    ✓ test adding ERC20TransferHandler that does not swap incorrectly and transferTokens reverts (445ms)
  Test cancel
    ✓ test cancellation with no items
    ✓ test cancellation with one item (54ms)
    ✓ test an array of nonces, ensure the cancellation of only those orders (140ms)
  Test cancelUpTo functionality
    ✓ test that given a minimum nonce for a signer is set (62ms)
    ✓ test that given a minimum nonce that all orders below a nonce value are cancelled
  Test authorize signer
    ✓ test when the message sender is the authorized signer
  Test revoke
    ✓ test that the revokeSigner is successfully removed (51ms)
    ✓ test that the revokeSender is successfully removed (49ms)

Contract: Swap
  Deploying...
    ✓ Deployed Swap contract (197ms)
    ✓ Deployed test contract "AST" (44ms)
    ✓ Deployed test contract "DAI" (43ms)
    ✓ Check that TransferHandlerRegistry correctly set
    ✓ Set up TokenRegistry and ERC20TransferHandler (64ms)
  Minting ERC20 tokens (AST and DAI)...
    ✓ Mints 1000 AST for Alice (77ms)
    ✓ Mints 1000 DAI for Bob (74ms)
  Approving ERC20 tokens (AST and DAI)...
    ✓ Checks approvals (Alice 250 AST and 0 DAI, Bob 0 AST and 1000 DAI) (134ms)
  Swaps (Fungible)
    ✓ Checks that Alice cannot swap more than balance approved (2000 AST for 50 DAI) (529ms)
    ✓ Checks that Bob can swap with Alice (200 AST for 50 DAI) (289ms)
    ✓ Checks that Alice cannot swap with herself (200 AST for 50 AST) (86ms)
    ✓ Alice sends Bob with an unknown kind for 10 DAI (474ms)
    ✓ Checks balances... (64ms)
    ✓ Checks that Bob cannot take the same order again (200 AST for 50 DAI) (50ms)
    ✓ Checks balances... (66ms)
    ✓ Checks that Alice cannot trade more than approved (200 AST) (98ms)
    ✓ Checks that Bob cannot take an expired order (46ms)
    ✓ Checks that an order is expired when expiry == block.timestamp (52ms)
    ✓ Checks that Bob can not trade more than he holds (82ms)
    ✓ Checks remaining balances and approvals (212ms)
    ✓ Checks that adding an affiliate address but empty token address and amount still swaps (347ms)
    ✓ Transfers tokens back for future tests (304ms)
Signer Delegation (Signer-side)
    ✓ Checks that David cannot make an order on behalf of Alice (85ms)
    ✓ Checks that David cannot make an order on behalf of Alice without signature (82ms)
    ✓ Alice attempts to incorrectly authorize herself to make orders
    ✓ Alice authorizes David to make orders on her behalf
    ✓ Alice authorizes David a second time does not emit an event
    ✓ Alice approves Swap to spend the rest of her AST
    ✓ Checks that David can make an order on behalf of Alice (314ms)
    ✓ Alice revokes authorization from David (38ms)
    ✓ Alice fails to try to revokes authorization from David again
    ✓ Checks that David can no longer make orders on behalf of Alice (97ms)
    ✓ Checks remaining balances and approvals (286ms)
Sender Delegation (Sender-side)
    ✓ Checks that Carol cannot take an order on behalf of Bob (69ms)
    ✓ Bob tries to unsuccessfully authorize himself to be an authorized sender
    ✓ Bob authorizes Carol to take orders on his behalf
```

```
    ✓ Bob authorizes Carol a second time does not emit an event
    ✓ Checks that Carol can take an order on behalf of Bob (308ms)
    ✓ Bob revokes sender authorization from Carol
    ✓ Bob fails to revoke sender authorization from Carol a second time
    ✓ Checks that Carol can no longer take orders on behalf of Bob (66ms)
    ✓ Checks remaining balances and approvals (205ms)
Signer and Sender Delegation (Three Way)
    ✓ Alice approves David to make orders on her behalf (50ms)
    ✓ Bob approves David to take orders on his behalf (38ms)
    ✓ Alice gives an unsigned order to David who takes it for Bob (199ms)
    ✓ Checks remaining balances and approvals (160ms)
Signer and Sender Delegation (Four Way)
    ✓ Bob approves Carol to take orders on his behalf
    ✓ David makes an order for Alice, Carol takes the order for Bob (345ms)
    ✓ Bob revokes the authorization to Carol
    ✓ Checks remaining balances and approvals (141ms)
Cancels
    ✓ Checks that Alice is able to cancel order with nonce 1
    ✓ Checks that Alice is unable to cancel order with nonce 1 twice
    ✓ Checks that Bob is unable to take an order with nonce 1 (46ms)
    ✓ Checks that Alice is able to set a minimum nonce of 4
    ✓ Checks that Bob is unable to take an order with nonce 2 (50ms)
    ✓ Checks that Bob is unable to take an order with nonce 3 (58ms)
    ✓ Checks existing balances (Alice 650 AST and 180 DAI, Bob 350 AST and 820 DAI) (146ms)
  Swaps with Fees
    ✓ Checks that Carol gets paid 50 AST for facilitating a trade between Alice and Bob (410ms)
    ✓ Checks balances... (97ms)
  Swap with Public Orders (No Sender Set)
    ✓ Checks that a Swap succeeds without a sender wallet set (292ms)
  Signatures
    ✓ Checks that an invalid signer signature will revert (328ms)
    ✓ Alice authorizes Eve to make orders on her behalf
    ✓ Checks that an invalid delegate signature will revert (376ms)
    ✓ Checks that an invalid signature version will revert (146ms)
    ✓ Checks that a private key signature is valid (285ms)
    ✓ Checks that a typed data (EIP712) signature is valid (294ms)


  131 passing (23s)

lerna info run Ran npm script 'test' in '@airswap/delegate' in 29.7s:
$ truffle test
Using network 'development'.


Compiling your contracts...
===========================
> Compiling ./contracts/interfaces/IDelegate.sol

    > compilation warnings encountered:

@airswap/types/contracts/Types.sol:18:1: Warning: Experimental features are turned on. Do not use experimental
features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^
,/Users/ezulkosk/audits/airswap-protocols/source/delegate/contracts/interfaces/IDelegate.sol:18:1: Warning:
Experimental features are turned on. Do not use experimental features on live deployments.
pragma experimental ABIEncoderV2;
^-------------------------------^



  Contract: Delegate Factory Tests
    Test deploying factory
      ✓ should have set swapContract
      ✓ should have set indexerContract
    Test deploying delegates
      ✓ should emit event and update the mapping (135ms)
      ✓ should create delegate with the correct values (361ms)

  Contract: Delegate Unit Tests
    Test constructor
      ✓ Test initial Swap Contract
      ✓ Test initial trade wallet value
      ✓ Test initial protocol value
      ✓ Test constructor sets the owner as the trade wallet on empty address (193ms)
      ✓ Test owner is set correctly having been provided an empty address
      ✓ Test owner is set correctly if provided an address (180ms)
      ✓ Test indexer is unable to pull funds from delegate account (258ms)
    Test setRule
      ✓ Test setRule permissions as not owner (76ms)
      ✓ Test setRule permissions as owner (121ms)
      ✓ Test setRule (98ms)
      ✓ Test setRule for zero priceCoef does revert (62ms)
    Test unsetRule
      ✓ Test unsetRule permissions as not owner (89ms)
      ✓ Test unsetRule permissions (54ms)
      ✓ Test unsetRule (167ms)
    Test setRuleAndIntent()
      ✓ Test calling setRuleAndIntent with transfer error (589ms)
      ✓ Test successfully calling setRuleAndIntent with 0 staked amount (260ms)
      ✓ Test successfully calling setRuleAndIntent with staked amount (312ms)
      ✓ Test unsuccessfully calling setRuleAndIntent with decreased staked amount (998ms)
    Test unsetRuleAndIntent()
      ✓ Test calling unsetRuleAndIntent() with transfer error (466ms)
      ✓ Test successfully calling unsetRuleAndIntent() with 0 staked amount (179ms)
      ✓ Test successfully calling unsetRuleAndIntent() with staked amount (515ms)
      ✓ Test successfully calling unsetRuleAndIntent() with staked amount (427ms)
    Test setTradeWallet
      ✓ Test setTradeWallet when not owner (81ms)
      ✓ Test setTradeWallet when owner (148ms)
      ✓ Test setTradeWallet with empty address (88ms)
```

```
Test transfer of ownership
   ✓ Test ownership after transfer (103ms)
Test getSignerSideQuote
   ✓ test when rule does not exist
   ✓ test when delegate amount is greater than max delegate amount (88ms)
   ✓ test when delegate amount is 0 (102ms)
   ✓ test a successful call - getSignerSideQuote (91ms)
Test getSenderSideQuote
   ✓ test when rule does not exist (64ms)
   ✓ test when delegate amount is not within acceptable value bounds (104ms)
   ✓ test a successful call - getSenderSideQuote (68ms)
Test getMaxQuote
   ✓ test when rule does not exist
   ✓ test a successful call - getMaxQuote (67ms)
Test provideOrder
   ✓ test if a rule does not exist (84ms)
   ✓ test if an order exceeds maximum amount (120ms)
   ✓ test if the sender is not empty and not the trade wallet (107ms)
   ✓ test if order is not priced according to the rule (118ms)
   ✓ test if order sender and signer amount are not matching (191ms)
   ✓ test if order signer kind is not an ERC20 interface id (110ms)
   ✓ test if order sender kind is not an ERC20 interface id (123ms)
   ✓ test a successful transaction with integer values (310ms)
   ✓ test a successful transaction with trade wallet as sender (278ms)
   ✓ test a successful transaction with decimal values (253ms)
   ✓ test a getting a signerSideQuote and passing it into provideOrder (241ms)
   ✓ test a getting a senderSideQuote and passing it into provideOrder (252ms)
   ✓ test a getting a getMaxQuote and passing it into provideOrder (252ms)
   ✓ test the signer trying to trade just 1 unit over the rule price - fails (121ms)
   ✓ test the signer trying to trade just 1 unit less than the rule price - passes (212ms)
   ✓ test the signer trying to trade the exact amount of rule price - passes (269ms)
   ✓ Send order without signature to the delegate (55ms)

Contract: Delegate Integration Tests
  Test the delegate constructor
     ✓ Test that delegateOwner set as 0x0 passes (87ms)
     ✓ Test that trade wallet set as 0x0 passes (83ms)
  Checks setTradeWallet
     ✓ Does not set a 0x0 trade wallet (41ms)
     ✓ Does set a new valid trade wallet address (80ms)
     ✓ Non-owner cannot set a new address (42ms)
  Checks set and unset rule
     ✓ Set and unset a rule for WETH/DAI (123ms)
     ✓ Test setRule for zero priceCoef does revert (52ms)
  Test setRuleAndIntent()
     ✓ Test successfully calling setRuleAndIntent (345ms)
     ✓ Test successfully increasing stake with setRuleAndIntent (312ms)
     ✓ Test successfully decreasing stake to 0 with setRuleAndIntent (313ms)
     ✓ Test successfully calling setRuleAndIntent (318ms)
     ✓ Test successfully calling setRuleAndIntent with no-stake change (196ms)
Test unsetRuleAndIntent()
   ✓ Test successfully calling unsetRuleAndIntent() (223ms)
   ✓ Test successfully setting stake to 0 with setRuleAndIntent and then unsetting (402ms)
Checks pricing logic from the Delegate
   ✓ Send up to 100K WETH for DAI at 300 DAI/WETH (76ms)
   ✓ Send up to 100K DAI for WETH at 0.0032 WETH/DAI (71ms)
   ✓ Send up to 100K WETH for DAI at 300.005 DAI/WETH (110ms)
Checks quotes from the Delegate
   ✓ Gets a quote to buy 23412 DAI for WETH (Quote: 74.9184 WETH)
   ✓ Gets a quote to sell 100K (Max) DAI for WETH (Quote: 320 WETH)
   ✓ Gets a quote to sell 1 WETH for DAI (Quote: 312.5 DAI)
   ✓ Gets a quote to sell 500 DAI for WETH (False: No rule)
   ✓ Gets a max quote to buy WETH for DAI
   ✓ Gets a max quote for a non-existent rule (100ms)
   ✓ Gets a quote to buy WETH for 250000 DAI (False: Exceeds Max)
   ✓ Gets a quote to buy 500 WETH for DAI (False: Exceeds Max)
Test tradeWallet logic
   ✓ should not trade for a different wallet (72ms)
   ✓ should not accept open trades (65ms)
   ✓ should not trade if the tradeWallet hasn't authorized the delegate to send (290ms)
   ✓ should not trade if the tradeWallet's authorization has been revoked (327ms)
   ✓ should trade if the tradeWallet has authorized the delegate to send (601ms)
Provide some orders to the Delegate
   ✓ Use quote with non-existent rule (90ms)
   ✓ Send order without signature to the delegate (107ms)
   ✓ Use quote larger than delegate rule (117ms)
   ✓ Use incorrect price on delegate (78ms)
   ✓ Use quote with incorrect signer token kind (80ms)
   ✓ Use quote with incorrect sender token kind (93ms)
   ✓ Gets a quote to sell 1 WETH and takes it, swap fails (275ms)
   ✓ Gets a quote to sell 1 WETH and takes it, swap passes (463ms)
   ✓ Gets a quote to sell 1 WETH where sender != signer, passes (475ms)
   ✓ Queries signerSideQuote and passes the value into an order (567ms)
   ✓ Queries senderSideQuote and passes the value into an order (507ms)
   ✓ Queries getMaxQuote and passes the value into an order (613ms)


  98 passing (22s)

lerna info run Ran npm script 'test' in '@airswap/debugger' in 12.3s:
$ mocha test --timeout 3000 --exit


  Orders
     ✓ Check correct order without signature (1281ms)
     ✓ Check correct order with signature (991ms)
     ✓ Check expired order (902ms)
     ✓ Check invalid signature (816ms)
     ✓ Check order without allowance (1070ms)
     ✓ Check NFT order without balance or allowance (1080ms)
     ✓ Check invalid token kind (654ms)
     ✓ Check NFT order without allowance (1045ms)
     ✓ Check NFT order to an invalid contract (1196ms)
```

```
      ✓ Check NFT order to a valid contract (1225ms)
      ✓ Check order without balance (839ms)


  11 passing (11s)

lerna info run Ran npm script 'test' in '@airswap/pre-swap-checker' in 11.6s:
$ truffle test
Using network 'development'.


Compiling your contracts...
===========================
> Everything is up to date, there is nothing to compile.



  Contract: PreSwapChecker
    Deploying...
       ✓ Deployed Swap contract (217ms)
       ✓ Deployed SwapChecker contract
       ✓ Deployed test contract "AST" (56ms)
       ✓ Deployed test contract "DAI" (40ms)
    Minting...
       ✓ Mints 1000 AST for Alice (76ms)
       ✓ Mints 1000 DAI for Bob (78ms)
    Approving...
       ✓ Checks approvals (Alice 250 AST and 0 DAI, Bob 0 AST and 500 DAI) (186ms)
    Swaps (Fungible)
       ✓ Checks fillable order is empty error array (517ms)
       ✓ Checks that Alice cannot swap with herself (200 AST for 50 AST) (296ms)
       ✓ Checks error messages for invalid balances and approvals (236ms)
       ✓ Checks filled order emits error (641ms)
       ✓ Checks expired, low nonced, and invalid sig order emits error (315ms)
       ✓ Alice authorizes Carol to make orders on her behalf (38ms)
       ✓ Check from a different approved signer and empty sender address (271ms)
    Deploying non-fungible token...
       ✓ Deployed test contract "Collectible" (49ms)
    Minting and testing non-fungible token...
       ✓ Mints a kitty collectible (#54321) for Bob (55ms)
       ✓ Alice tries to buys non-owned Kitty #54320 from Bob for 50 AST causes revert (97ms)
       ✓ Alice tries to buys non-approved Kitty #54321 from Bob for 50 AST (192ms)


  18 passing (4s)

lerna info run Ran npm script 'test' in '@airswap/wrapper' in 22.7s:
$ truffle test
Using network 'development'.

Compiling your contracts...
===========================
> Everything is up to date, there is nothing to compile.



  Contract: Wrapper Unit Tests
    Test initial values
       ✓ Test initial Swap Contract
       ✓ Test initial Weth Contract
       ✓ Test fallback function revert
    Test swap()
       ✓ Test when sender token != weth, ensure no unexpected ether sent (78ms)
       ✓ Test when sender token == weth, ensure the sender amount matches sent ether (119ms)
       ✓ Test when sender token == weth, signer token == weth, and the transaction passes (362ms)
       ✓ Test when sender token == weth, signer token != weth, and the transaction passes (243ms)
       ✓ Test when sender token == weth, signer token != weth, and the wrapper token transfer fails (122ms)
    Test swap() with two ERC20s
       ✓ Test when sender token == non weth erc20, signer token == non weth erc20 but msg.sender is not
senderwallet (55ms)
       ✓ Test when sender token == non weth erc20, signer token == non weth erc20, and the transaction passes
(172ms)
    Test provideDelegateOrder()
       ✓ Test when signer token != weth, but unexpected ether sent (84ms)
       ✓ Test when signer token == weth, but no ether is sent (101ms)
       ✓ Test when signer token == weth, but no signature is sent (54ms)
       ✓ Test when signer token == weth, but incorrect amount of ether sent (63ms)
       ✓ Test when signer token == weth, correct eth sent, tx passes (247ms)
       ✓ Test when signer token != weth, sender token == weth, wrapper cant transfer eth (506ms)
       ✓ Test when signer token != weth, sender token == weth, tx passes (291ms)

  Contract: Wrapper
    Setup
       ✓ Mints 1000 DAI for Alice (49ms)
       ✓ Mints 1000 AST for Bob (57ms)
    Approving...
       ✓ Alice approves Swap to spend 1000 DAI (40ms)
       ✓ Bob approves Swap to spend 1000 AST
       ✓ Bob approves Swap to spend 1000 WETH
       ✓ Bob authorizes the Wrapper to send orders on his behalf
    Test swap(): Wrap Buys
       ✓ Checks that Bob take a WETH order from Alice using ETH (513ms)
    Test swap(): Unwrap Sells
       ✓ Carol gets some WETH and approves on the Swap contract (64ms)
       ✓ Alice authorizes the Wrapper to send orders on her behalf
       ✓ Alice approves the Wrapper contract to move her WETH
       ✓ Checks that Alice receives ETH for a WETH order from Carol (460ms)
    Test swap(): Sending ether and WETH to the WrapperContract without swap issues
       ✓ Sending ether to the Wrapper Contract
       ✓ Sending WETH to the Wrapper Contract (70ms)
       ✓ Alice approves Swap to spend 1000 DAI
       ✓ Send order where the sender does not send the correct amount of ETH (69ms)
```

```
    ✓ Send order where Bob sends Eth to Alice for DAI (422ms)
    ✓ Reverts if the unwrapped ETH is sent to a non-payable contract (1117ms)
Test swap(): Sending nonWETH ERC20
    ✓ Alice approves Swap to spend 1000 DAI (38ms)
    ✓ Bob approves Swap to spend 1000 AST
    ✓ Send order where Bob sends AST to Alice for DAI (447ms)
    ✓ Send order where the sender is not the sender of the order (100ms)
    ✓ Send order without WETH where ETH is incorrectly supplied (70ms)
    ✓ Send order where Bob sends AST to Alice for DAI w/ authorization but without signature (48ms)
Test provideDelegateOrder()
  Wrap Buys
      ✓ Check Carol sending no ETH with order (66ms)
      ✓ Check Carol not signing order (46ms)
      ✓ Check Carol sets the wrong sender wallet (217ms)
      ✓ Check delegate owner hasnt authorised the delegate as sender swap (390ms)
      ✓ Check Carol hasnt given swap approval to swap WETH (906ms)
      ✓ Check successful ETH wrap and swap through delegate contract (575ms)
  Unwrap Sells
      ✓ Check Carol sending ETH when she shouldnt (64ms)
      ✓ Check Carol not signing the order (42ms)
      ✓ Check Carol hasnt given swap approval to swap DAI (1043ms)
      ✓ Check Carol doesnt approve wrapper to transfer weth (951ms)
      ✓ Check successful ETH wrap and swap through delegate contract (646ms)

  51 passing (15s)

lerna success run Ran npm script 'test' in 9 packages in 155.7s:
lerna success - @airswap/delegate
lerna success - @airswap/indexer
lerna success - @airswap/swap
lerna success - @airswap/transfers
lerna success - @airswap/types
lerna success - @airswap/wrapper
lerna success - @airswap/debugger
lerna success - @airswap/order-utils
lerna success - @airswap/pre-swap-checker
 ✓  Done in 222.01s.
```

Code Coverage

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	100	100	100	100	
Delegate.sol	100	100	100	100	
Imports.sol	100	100	100	100	
contracts/interfaces/	100	100	100	100	
IDelegate.sol	100	100	100	100	
All files	100	100	100	100	
File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	100	100	100	100	
DelegateFactory.sol	100	100	100	100	
Imports.sol	100	100	100	100	
contracts/interfaces/	100	100	100	100	
IDelegateFactory.sol	100	100	100	100	
All files	100	100	100	100	
File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	100	100	100	100	
Imports.sol	100	100	100	100	
Index.sol	100	100	100	100	
All files	100	100	100	100	
File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	100	100	100	100	
Indexer.sol	100	100	100	100	
contracts/interfaces/	100	100	100	100	
IIndexer.sol	100	100	100	100	
ILocatorWhitelist.sol	100	100	100	100	
All files	100	100	100	100	
File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	100	100	100	100	
Imports.sol	100	100	100	100	
Swap.sol	100	100	100	100	
All files	100	100	100	100	
File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	100	100	100	100	
Imports.sol	100	100	100	100	
Types.sol	100	100	100	100	
All files	100	100	100	100	
File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	100	100	100	100	
Imports.sol	100	100	100	100	
Wrapper.sol	100	100	100	100	
All files	100	100	100	100	

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

```
1c4e30fd3aa765cb0ee259a29dead71c1c99888dcc7157c25df3405802cf5b09 ./source/indexer/contracts/Migrations.sol

853f79563b2b4fba199307619ff5db6b86ceae675eb259292f752f3126c21236 ./source/indexer/contracts/Imports.sol

b7d114e1b96633016867229abb1d8298c12928bd6e6935d1969a3e25133d0ae3 ./source/indexer/contracts/Indexer.sol

cb278eb599018f2bcff3e7eba06074161f3f98072dc1f11446da6222111e6fb6
./source/indexer/contracts/interfaces/IIndexer.sol

ae69440b7cc0ebde7d315079effaaf6db840a5c36719e64134d012f183a82b3c
./source/indexer/contracts/interfaces/ILocatorWhitelist.sol

0ce96202a3788403e815bcd033a7c2528d2eceb9e6d0d21f58fd532e0721c3f3 ./source/tokens/contracts/WETH9.sol

f49af1f0a94dc5d58725b7715ff4a1f241626629aa68c442dd51c540f3b40ee7 ./source/tokens/contracts/NonFungibleToken.sol

13e6724efe593830fdd839337c2735a9bfbd6c72fc6134d9622b1f38da734fed ./source/tokens/contracts/IERC721Receiver.sol

b0baff5c036f01ac95dae20048f6ee4716796b293f066d603a2934bee8aa049f ./source/tokens/contracts/OrderTest721.sol

27ffdcc5bfb7e1352c69f56869a43db1b1d76ee9856fbfd817d6a3be3d378a89 ./source/tokens/contracts/FungibleToken.sol

1c4e30fd3aa765cb0ee259a29dead71c1c99888dcc7157c25df3405802cf5b09 ./source/tokens/contracts/Migrations.sol

a41dd3eaddff2d0ce61f9d0d2d774f57bf286ba7534fe7392cb331c52587f007 ./source/tokens/contracts/AdaptedERC721.sol

a497e0e9c84e431234f8ef23e5a3bb72e0a8d8e5381909cc9f274c6f8f0f8693 ./source/tokens/contracts/KittyCore.sol

afc8395fc3e20eb17d99ae53f63cae764250f5dda6144b0695c9eb3c29065daa ./source/tokens/contracts/OMGToken.sol

f07b61827716f0e44db91baabe9638eef66e85fb2d720e1b221ee03797eb0c16 ./source/tokens/contracts/interfaces/IWETH.sol

2f4455987f24caf5d69bd9a3c469b05350ec5b0cc62cc829109cfa07a9ed184c
./source/tokens/contracts/interfaces/INRERC20.sol

4deea5147ee8eedbeaf394454e63a32bce34003266358abb7637843eec913109 ./source/index/contracts/Index.sol

1c4e30fd3aa765cb0ee259a29dead71c1c99888dcc7157c25df3405802cf5b09 ./source/index/contracts/Migrations.sol

8304b9b7777834e7dd882ecef91c8376160da6a6dd6e4c7c9f9b99bb8a0ddb08 ./source/index/contracts/Imports.sol

93dbd794dd6bbc93c47a11dc734efb6690495a1d5138036ebee8687edeb25dc6 ./source/delegate-
factory/contracts/DelegateFactory.sol

1c4e30fd3aa765cb0ee259a29dead71c1c99888dcc7157c25df3405802cf5b09 ./source/delegate-
factory/contracts/Migrations.sol

d4641deca8ebf06d554ef39b9fe90f2db23f40be7557d8bbc5826428ba9b0d0a ./source/delegate-factory/contracts/Imports.sol

6371ccf87ef8e8cddfceab2903a109714ab5bcaaa72e7096bff9b8f0a7c643a9 ./source/delegate-
factory/contracts/interfaces/IDelegateFactory.sol

c3762a171563d0d2614da11c4c0e17a722aa2bc4a4efa126b54420a3d7e7e5b1 ./source/delegate/contracts/Migrations.sol

0843c702ea883afa38e874a9d16cb4830c3c8e6dadf324ddbe0f397dd5f67aeb ./source/delegate/contracts/Imports.sol

cbe7e7fa0e85995f86dde9f103897ac533a42c78ee017a49d29075adf5152be4 ./source/delegate/contracts/Delegate.sol

4ea8cec0ad9ff88426e7687384f5240d4444ee48407ddd07e39ab527ba70d94e
./source/delegate/contracts/interfaces/IDelegate.sol

c3762a171563d0d2614da11c4c0e17a722aa2bc4a4efa126b54420a3d7e7e5b1 ./source/swap/contracts/Migrations.sol

3d764b8d0ebb2aa5c765f0eb0ef111564af96813fd6427c39d8a11c4251cbba2 ./source/swap/contracts/Imports.sol

001573b0de147db510c6df653935505d7f0c3e24d0d5028ebfdffa06055b9508 ./source/swap/contracts/Swap.sol

b2693adaefd67dfcefd6e942aee9672469d0635f8c6b96183b57667e82f9be73 ./source/swap/contracts/interfaces/ISwap.sol

3d9797822cc119ac07cfb02705033d982d429ad46b0d39a0cfa4f7df1d9bfdd6 ./source/wrapper/contracts/HelperMock.sol

a0a4226ee86de0f2f163d9ca14e9486b9a9a82137e4e97495564cd37e92c8265 ./source/wrapper/contracts/Migrations.sol

853712933537f67add61f1299d0b763664116f3f36ae87f55743104fbc77861a ./source/wrapper/contracts/Imports.sol

67fc8542fb154458c3a6e4e07c3eb636f65ebb2074082ab24727da09b7684feb ./source/wrapper/contracts/Wrapper.sol

1c4e30fd3aa765cb0ee259a29dead71c1c99888dcc7157c25df3405802cf5b09 ./source/types/contracts/Migrations.sol

74947f792f6128dad955558044e7a2d13ecda4f6887cb85d9bf12f4e01423e6e ./source/types/contracts/Imports.sol

95f41e78212c566ea22441a6e534feae44b7505f65eea89bf67dc7a73910821e ./source/types/contracts/Types.sol

fe4fc1137e2979f1af89f69b459244261b471b5fdbd9bdbac74382c14e8de4db ./source/types/test/MockTypes.sol
```

Tests

```
82257690d4d4ac0e34082491115c1eb822d83d4aa6bcf6e752b5ef7db57e8cf7 ./source/indexer/truffle-config.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/indexer/coverage/prettify.js
```

```
a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/indexer/coverage/sorter.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/indexer/coverage/lcov-
report/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/indexer/coverage/lcov-report/sorter.js

9b9b6feb6ad0bf0d1c9ca2e89717499152d278ff803795ab25b975826ce3e6fb ./source/indexer/test/Indexer-unit.js

b8afaf2ac9876ada39483c662e3017288e78641d475c3aad8baae3e42f2692b1 ./source/indexer/test/Indexer.js

82257690d4d4ac0e34082491115c1eb822d83d4aa6bcf6e752b5ef7db57e8cf7 ./source/tokens/truffle-config.js

82257690d4d4ac0e34082491115c1eb822d83d4aa6bcf6e752b5ef7db57e8cf7 ./source/index/truffle-config.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/index/coverage/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/index/coverage/sorter.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/index/coverage/lcov-report/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/index/coverage/lcov-report/sorter.js

5b30ebaf2cb02fdb583a91b8d80e0d3332f613f1f9d03227fa1f497182612d79 ./source/index/test/Index-unit.js

82257690d4d4ac0e34082491115c1eb822d83d4aa6bcf6e752b5ef7db57e8cf7 ./source/delegate-factory/truffle-config.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/delegate-factory/coverage/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/delegate-factory/coverage/sorter.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/delegate-factory/coverage/lcov-
report/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/delegate-factory/coverage/lcov-
report/sorter.js

e1e96cac95b7ca35a3eff407e69378395fee64fbd40bc46731e02cab3386f1a9 ./source/delegate-factory/test/Delegate-
Factory-unit.js

82257690d4d4ac0e34082491115c1eb822d83d4aa6bcf6e752b5ef7db57e8cf7 ./source/delegate/truffle-config.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/delegate/coverage/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/delegate/coverage/sorter.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/delegate/coverage/lcov-
report/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/delegate/coverage/lcov-
report/sorter.js

0d39c455ec8b473be0aa20b21fff3324e87fe688281cc29ce446992682766197 ./source/delegate/test/Delegate.js

6568ea0ed57b6cfb6e9671ae07e9721e80b9a74f4af2a1f31a730df45eed7bc4 ./source/delegate/test/Delegate-unit.js

67a94a4b8a64b862eac78c2be9a76fdfb57412113b0e98342cf9be743c065045 ./source/swap/.solcover.js

82257690d4d4ac0e34082491115c1eb822d83d4aa6bcf6e752b5ef7db57e8cf7 ./source/swap/truffle-config.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/swap/coverage/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/swap/coverage/sorter.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/swap/coverage/lcov-report/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/swap/coverage/lcov-report/sorter.js

eb606ca3e7fe215a183e67c73af188a3a463a73a2571896403890bd6308b9553 ./source/swap/test/Swap.js

02ed0eee9b5fd1bdb2e29ef7c76902b8c7788d56cccb08ba0a1c62acdb0c240d ./source/swap/test/Swap-unit.js

82257690d4d4ac0e34082491115c1eb822d83d4aa6bcf6e752b5ef7db57e8cf7 ./source/wrapper/truffle-config.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/wrapper/coverage/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/wrapper/coverage/sorter.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/wrapper/coverage/lcov-
report/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/wrapper/coverage/lcov-report/sorter.js

8e8dcdef012cdc8bf9dc2758afcb654ce3ec55a4522ebab9d80455813c1c2234 ./source/wrapper/test/Wrapper.js

fb48b5abc2cc9cfd07767e93c37130ff412088741f0685deb74c65b1b596daf9 ./source/wrapper/test/Wrapper-unit.js

82257690d4d4ac0e34082491115c1eb822d83d4aa6bcf6e752b5ef7db57e8cf7 ./source/types/truffle-config.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/types/coverage/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/types/coverage/sorter.js

67126b6cd4d1b2305f8c8fa5974971ebe90ab2b0f6e209ba2f1c6e4af05f0207 ./source/types/coverage/lcov-report/prettify.js

a2e1ee8eb42ae6152ffb680f1f3419cf4a189412b4ffc663252492d47a968914 ./source/types/coverage/lcov-report/sorter.js

94e6cf001c8edb49546d881416a1755aabe0a01f562df4140be166c3af2936fc ./source/types/test/Types-unit.js
```

About Quantstamp

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

Quantstamp's team boasts decades of combined experience in formal verification, static analysis, and software verification. Collectively, our individuals have over 500 Google scholar citations and numerous published papers. In its mission to proliferate development and adoption of blockchain applications, Quantstamp is also developing a new protocol for smart contract verification to help smart contract developers and projects worldwide to perform cost-effective smart contract security audits.

To date, Quantstamp has helped to secure hundreds of millions of dollars of transaction value in smart contracts and has assisted dozens of blockchain projects globally with its white glove security auditing 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.

Finally, Quantstamp's dedication to research and development in the form of collaborations with leading academic institutions such as National University of Singapore and MIT (Massachusetts Institute of Technology) reflects Quantstamp's commitment to enable world-class smart contract innovation.