Quantstamp Security Assessment Certificate

August 11th 2022 — Quantstamp Verified

Oasys

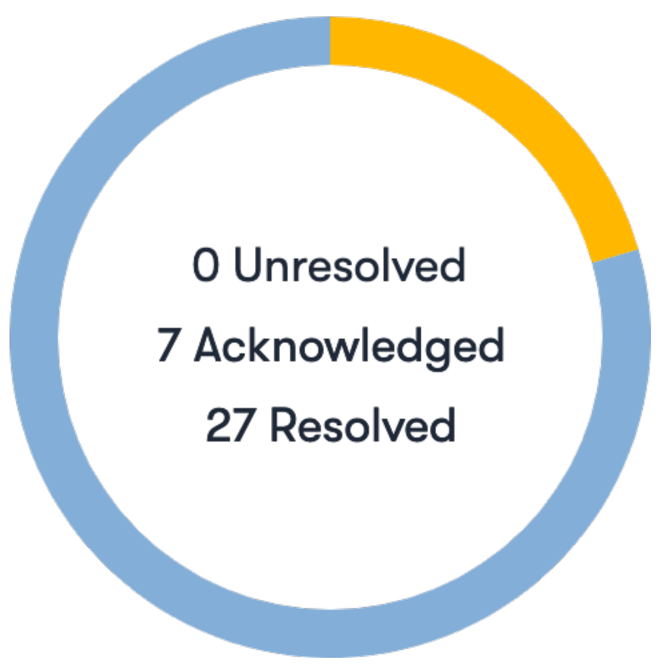
This audit report was prepared by Quantstamp, the leader in blockchain security.

Executive Summary

Type	Gaming Blockchain & Bridge Contracts
Auditors	David Knott, Senior Research Engineer Guillermo Escobero, Security Auditor Kacper Bąk, Senior Research Engineer Jan Gorzny, Blockchain Researcher Poming Lee, Senior Research Engineer
Timeline	2022-05-09 through 2022-07-01
EVM	Gray Glacier
Languages	Go, Solidity
Methods	Architecture Review, Unit Testing, Computer-Aided Verification, Manual Review
Specification	Oasys Technical Materials
Documentation Quality	Low
Test Quality	Medium
Source Code	

Repository	Commit
oasysgames/oasys-genesis-contract (initial)	3f00026
oasysgames/oasys-validator (initial)	ac3527f
oasysgames/oasys-optimism (initial)	134491c
oasysgames/oasys-genesis-contract (fixes)	1a5673f
oasysgames/oasys-validator (fixes)	e567281
oasysgames/oasys-optimism (fixes)	31d61c4

Total Issues	34	(27 Resolved)
High Risk Issues	11	(10 Resolved)
Medium Risk Issues	8	(5 Resolved)
Low Risk Issues	8	(8 Resolved)
Informational Risk Issues	4	(1 Resolved)
Undetermined Risk Issues	3	(3 Resolved)



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

Unresolved	Acknowledged the existence of the risk, and decided to accept it without engaging in special efforts to control it.
Acknowledged	The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).
Fixed	Adjusted program implementation, requirements or constraints to eliminate the risk.
Mitigated	Implemented actions to minimize the impact or likelihood of the risk.

Summary of Findings

After initial audit: Quantstamp has performed a full audit of Oasys' Genesis Contracts and a partial difference audit (not all modified/added files were in the audit scope) between Oasys' Validator Client and Go Ethereum which it is based off of. Notably, we found the project to have minimal fault tolerance due to its highly configurable consensus parameters. Additionally, some significant issues were found, including a lack of integration and unit tests, sparse documentation for all the audited functionalities, and signatures checks for privileged functions being incorrectly implemented. We recommend that all issues reported in this document be addressed.
Update (after fixes): Quantstamp has performed a re-audit of Oasys' Genesis Contracts and a partial difference re-audit (not all modified/added files were in the audit scope) between Oasys' Validator Client and Go Ethereum which it is based off of. Quantstamp also asked a number of questions to the Oasys team to clarify the audit findings and gain context on the intentions behind Oasys' teams intended initial `OAS` supply, `validator` rotation, and signature replayability design decisions. Though all issues were addressed the Oasys network is still highly centralized. Furthermore, although no more issues were found in the re-audit, Quantstamp is still concerned about the lack of integration tests on Oasys' Validator Client.

ID	Description	Severity	Status
QSP-1	Missing Input Validation	High	Mitigated
QSP-2	Expected Validator For Producing Blocks Could be Slashed	High	Fixed
QSP-3	Specified Total Supply Violation	High	Fixed
QSP-4	Out Of Date Validator Client Smart Contracts	High	Fixed
QSP-5	Missing Tests for Validator Client	High	Mitigated
QSP-6	Claim Time Period Violation	High	Fixed
QSP-7	Gas Usage / Loop Concerns	High	Fixed
QSP-8	Signature Replay	High	Fixed
QSP-9	Wrong Function Selector in Signature Verification	High	Fixed
QSP-10	Users May Not Be Able to Withdraw Funds in Early Stages	High	Fixed
QSP-11	Missing Specification	High	Acknowledged
QSP-12	Validator Allowlist Introduces A Single Point Of Failure	Medium	Acknowledged
QSP-13	Poor Inline Code Documentation	Medium	Acknowledged
QSP-14	Allowlist Unbound Iteration Denial of Service	Medium	Fixed
QSP-15	Blocks Proposed By Unexpected Validators Accepted	Medium	Acknowledged
QSP-16	Predictable Schedule of Block Producers	Medium	Mitigated
QSP-17	Signers Count Can Go Below Threshold	Medium	Fixed
QSP-18	Signed Message May Be Replayed Across Chains	Medium	Fixed
QSP-19	Missing Tests for NFT Bridge Contracts	Medium	Fixed
QSP-20	Unlocked Pragma	Low	Fixed
QSP-21	Allowance Double-Spend Exploit	Low	Mitigated
QSP-22	Signature Malleability	Low	Fixed
QSP-23	Privileged Roles and Ownership	Low	Mitigated
QSP-24	Signers May Be Duplicated	Low	Fixed
QSP-25	Ownership Can Be Renounced	Low	Fixed
QSP-26	Ownership Can Be Transferred Bypassing Chain Verification	Low	Fixed
QSP-27	Signatures Valid Indefinetly	Low	Fixed
QSP-28	Unused Imports	Informational	Fixed
QSP-29	Ownership Can Be Renounced	Informational	Acknowledged
QSP-30	Allowance Double-Spend Exploit	Informational	Acknowledged
QSP-31	Transaction Ordering Dependence for `initialize()` Functions	Informational	Acknowledged
QSP-32	Calculations Based on Current Environment Value Parameters	Undetermined	Fixed
QSP-33	ERC721 Token May Be Blocked when Finalizing Withdrawal	Undetermined	Fixed
QSP-34	ERC721 Token May Be Blocked when Finalizing Deposit	Undetermined	Fixed

Quantstamp Audit Breakdown

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

DISCLAIMER:
This report only covers the files under the audit scope. In the `oasys-optimism` repository, only `L1BuildDeposit.sol` file was requested to audit. Please refer to the appendix section "File Signatures" for the full list of audited files and their digest values.

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

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

Methodology

The Quantstamp auditing process follows a routine series of steps:

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

Toolset

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

Setup

Tool Setup:

- Slither v0.8.3

Steps taken to run the tools:

1. Install the Slither tool: `pip3 install slither-analyzer`
2. Run Slither from the project directory: `slither .`(`packages/contracts` for `oasys-optimism`).

Findings

QSP-1 Missing Input Validation

Severity: *High Risk*

Status: Mitigated

File(s) affected: `contracts/token/SOAS.sol`, `contracts/StakeManager.sol`, `contracts/lib/EnvironmentValue.sol`, `contracts/lib/Allowlist.sol`, `contracts/nft-bridge/NFTBridgeRelayer.sol`, `contracts/nft-bridge/SidechainERC721.sol`, `contracts/nft-bridge/Signers.sol`, `packages/contracts/contracts/oasys/L1/L1BuildDeposit.sol`

Related Issue(s): SWC-123

Description: Usually missing input validation is classified as low severity. However, the insufficient validation of submitted `EnvironmentValues` may lead to a chain halting overflow. Therefore, it has been escalated to a high severity issue.
It is important to validate inputs to both protect against malicious parties and to avoid human error. Specifically, the following function's arguments are missing input validation:

1. **Acknowledged:** `SOAS.constructor(...)` function does not check whether `_staking` is a contract address.

2. **Fixed:** SOAS.mint(...) function does not check whether to is address(0) and _staking is a contract address.
3. **Fixed:** SOAS.claim(...) function does not check whether the claim amount is zero.
4. **Fixed:** SOAS.renounce(...) function does not check whether the renounce amount is zero.
5. **Acknowledged:** SOAS._beforeTokenTransfer(...) function does not check whether the to address is the SOAS contract address.
6. **Acknowledged:** StakeManager.initialize(...) function does not check that _environment and _allowlist are contract addresses.
7. **Fixed:**StakeManager.updateValidatorBlocks(...) does not check that the operators and counts lists have the same length.
8. **Acknowledged:** EnvironmentValue.validate(...) function does not check that the value being added will not cause overflows.
9. **Fixed:** Allowlist.addAddress(...) function does not check whether _address is address(0).
10. **Acknowledged:** L1BuildDeposit.constructor(...) does not check that:
    - _requiredAmount is greater than zero.
    - _lockedBlock is greater than zero.
    - _allowlist is a contract address.
11. **Acknowledged:** L1BuildDeposit.initialize(...) does not check that _agentAddress is not address(0).
12. **Fixed:** L1BuildDeposit.deposit(...) does not check that:
    - msg.value is greater than zero.
    - _builder is not address(0).
13. **Fixed:** L1BuildDeposit.withdraw(...) does not check that:
    - amount is greater than zero.
    - _builder is not address(0).
14. **Fixed:** Signers.constructor(...) does not check that:
    - _signers contains no duplicate addresses.
    - no _signers are address(0).
    - _threshold is greater than zero.
15. **Fixed:** Signers.verifySignatures(...) does not check that:
    - _hash is not an empty bytes32 string.
    - signatures is divisible by 65.
16. **Fixed:** Signers.addSigner(...) does not check that:
    - _address is not address(0).
17. **Fixed:** Signers.updateThreshold(...) does not check that _threshold is greater than zero.
18. **Fixed:** NFTBridgeRelayer.finalizeWithdrawal(...) does not check that mainTo is not address(0).
19. **Acknowledged:** NFTBridgeRelayer.transferMainchainRelayer(...) does not check that newRelayer is not address(0).
20. **Acknowledged:** NFTBridgeRelayer.createSidechainERC721(...) does not check that:
    - name is not an empty bytes string.
    - symbol is not an empty bytes string.
    - sideTo is not address(0).
21. **Acknowledged:** NFTBridgeRelayer.transferSidechainRelayer(...) does not check that newRelayer is not address(0).

Recommendation: We recommend adding the missing checks that were enumerated above.

Update: The Oasys team fixed issues 1.2, 1.3, 1.4, 1.7, 1.9, 1.12, 1.13, 1.14, 1.15, 1.16, 1.17 and 1.18, acknowledged issues 1.1, 1.5, 1.6, 1.8, 1.10, 1.11, 1.19, 1.20 and 1.21 with the following comments:
1. SOAS's constructor function does not check whether _staking is a contract address as the constructor is intended to be used for testing purposes only.
5. SOAS's _beforeTokenTransfer does not check whether the to address is the SOAS contract address as the check was deemed unnecessary.
6. StakeManager's initialize function does not check whether _environment and _allowlist are contract addresses as they are intended to be initialize programmatically by the Oasys Validator client.
8. Overflow checks were not added to EnvironmentValue's validate function as consensus changes require a governance vote, changes to github, and 51% of validators to upload the consensus changes. It is during these steps that potentially overflows are intended to be detected.
10. The Oasys team states that L1BuildDeposit.constructor(...) does not need input validation as it is used for testing purposes only. 11. The Oasys team states that L1BuildDeposit.initialize(...) is for test execution and that the process of setting is hardcoded in the Oasys Validator Client. 19. The Oasys team states: *No action is required as it is checked by* Ownable.transferOwnership(...). 20. The Oasys team states that name and symbol do not need to be checked for empty strings as they are allowed in their specification.
21. The Oasys team states: *No action is required as it is checked by* Ownable.transferOwnership(...).

Fixed on commit oasys-optimism@0749875, oasys-genesis-contract@18fbba0 and oasys-genesis-contract@ffa905a.

QSP-2 Expected Validator For Producing Blocks Could be Slashed

Severity: *High Risk*

Status: Fixed

File(s) affected: consensus/oasys/oasys.go

Description: In consensus/oasys/oasys.go on L657 the code only checks that validator != schedule[number] instead of checking whether the validator is an expected alternate validator according to the schedule when the original validator (i.e., schedule[number]) is down.

Recommendation: Check whether the validator proposing the block out of turn is an expected alternate. If the block proposer is an expected alternate, then slashing should be skipped.

Update: Response from the Oasys team: *There is no problem in thrashing the "expectedValidator" since it is a validator that should create the block in question.*

QSP-3 Specified Total Supply Violation

Severity: *High Risk*

Status: Fixed

File(s) affected: `consensus/oasys/oasys.go`, `genesis/mainnet.json`, `contracts/StakeManager.sol`

Description: The specified total supply is `10` billion but the initial total supply in `genesis/mainnet.json` is `7` billion. Additionally, `consensus/oasys/oasys.go` distributes rewards by increasing the `StakeManager` contract's balance by calling `state.AddBalance`. Calls to `state.AddBalance` without subtracting balance from another account increase the total supply of `OAS` which violates the specification that the total supply of OAS will be fixed for `6` years.

Recommendation: Modify either `genesis/mainnet.json`'s initial total supply or the stated initial total supply so that they match one another.
Modify the specification which states that the total supply of `OAS` will be fixed for the first `6` years or remove the call to `state.AddBalance`.

Update: The Oasys team states that `genesis/mainnet.json`'s initial balance is intended to be `7` billion and not `10` billion but that the total supply will not exceed `10` billion for the first six years after which it may depending on on-chain governance.

QSP-4 Out Of Date Validator Client Smart Contracts

Severity: *High Risk*

Status: Fixed

File(s) affected: `consensus/oasys/contract.go`, `contracts/Environment.sol`, `contracts/StakeManager.sol`

Description: `Environment`'s and `StakeManager`'s ABIs and bytecode do not match the audited smart contract code. In particular, support for `StakeManager.initialize(...)` function's `_allowlist` parameter is not included. This renders `StakeManager`'s validator functionality unusable as all calls to `allowlist` will be to `address(0)` and will revert.

Recommendation: We recommend setting up an automated system to keep your Go Validator Client's smart contract's ABIs and bytecode up to date with your smart contract code.

Update: Fixed on oasys-validator@4a0e701.

QSP-5 Missing Tests for Validator Client

Severity: *High Risk*

Status: Mitigated

File(s) affected: `All Go Files`

Description: The Oasys team forked Go Ethereum and made significant changes to it, turning it into the Oasys blockchain Validator Client. However, no tests were added to ensure that the changes made behave as expected.

Recommendation: Given the complexity of Go Ethereum and the intricate nature of the changes made, unit tests should be added for all changes made to Go Ethereum. Additionally, integration tests should be added to ensure that the unmodified parts of Go Ethereum still behave as expected when used to validate the Oasys blockchain.

Update: The Oasys team added unit tests and an integration test to their Go Ethereum changes though further testing is still recommended. Related commit: oasys-validator@3f6e5e9.

QSP-6 Claim Time Period Violation

Severity: *High Risk*

Status: Fixed

File(s) affected: `contracts/token/SOAS.sol`

Description: A reentrancy exploit can occur when external contract calls are made. To protect against reentrancy, it is recommended to order one's functions by:
Checks - Perform validation checks. Effects - Update all contract state. Interactions - Make external contract calls.
The above ordering protects one's functions from reentrancy because by the time external contract calls are made, and execution is given to a potentially untrusted contract, all contract state is already in the most up-to-date state.
`SOAS`'s `claim` function subtracts `claimInfo[msg.sender].claimed`, the amount a caller has already claimed, to determine how much a caller has left to claim. A reentrancy attack is possible because the amount a caller has already claimed is updated after an external contract call, transferring the claimed funds, has been made. This allows the `caller` to convert their entire `SOAS` balance to `OAS` in the time period between `info.since` and `info.until` instead of having to convert their `SOAS` to `OAS` gradually as is intended by the protocol.

Exploit Scenario: 1. Attacker deploys a malicious smart contract which has a `receive` function that calls SOAS's `claim` N times.

1. The attacker contract is minted `SOAS` with a `since` and `until` in the future.
2. Once the `since` time passes the attacker calls their malicious contract which calls `claim` repeatedly, converting all its `SOAS` back to `OAS` in a single transaction.

Recommendation: Move the incrementation of `claimInfo[msg.sender].claimed` on `L88` to be before the external contract call transferring `OAS` on `L85` to be inline with the checks/effect/interaction pattern and protect against reentrancy.

Update: Fixed on oasys-validator@ada3419.

QSP-7 Gas Usage / Loop Concerns

Severity: *High Risk*

Status: Fixed

File(s) affected: `contracts/StakeManager.sol`

Related Issue(s): SWC-126, SWC-134

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 loop requires too much gas to finish processing, then it may prevent the contract from functioning correctly entirely.
`StakeManager.updateValidators(...)` iterates over `currentValidators`, performing multiple `SLOAD`s and `SSTORE`s for each validator. For example with `100` validators at `4096` epochs, this operation could cost `6.5` million gas per epoch.

```
2100 (SLOAD cost) * 2**12 (isCandidate binary search worst case) + 20000 (SSTORE changing value on L236) + 20000 (SSTORE adding new value on L236)
= 65200 (gas cost per validator)
65200 * 100 (maximum number of validators)
= 6520000
```

Given that: (i) the `validatorThreshold` is configurable, (ii) the total supply of OAS is inflationary, and the number of epochs may be much greater than `4,096` the gas costs required to execute `updateValidators` may at some point exceed the Oasys blockchains block gas limit.

Recommendation: Modify all operations that iterate through validators or operators to be run be runnable over multiple blocks. Given the high gas costs associated with the `validatorUpdate` operation in particular, benchmark the gas costs of the worst case scenario to ensure that the gas cost of calling `updateValidators` will always be under the block gas limit.

Update: Fixed on oasys-genesis-contract@9bf50ba and oasys-validator@a52bea8.

QSP-8 Signature Replay

Severity: *High Risk*

Status: Fixed

File(s) affected: `contracts/nft-bridge/NFTBridgeRelayer.sol`, `contracts/nft-bridge/Signers.sol`

Related Issue(s): SWC-121

Description: Although the contract checks the validity of the signed message in `verifySignatures(...)`, the hashed value excludes a nonce, and, therefore, the signed message may be replayed.
The following functions could be affected:

- `Signers.addSigner(...)`
- `Signers.removeSigner(...)`
- `Signers.updateThreshold(...)`
- `NFTBridgeRelayer.transferMainchainRelayer(...)`
- `NFTBridgeRelayer.transferSidechainRelayer(...)`

Exploit Scenario:

1. Alice, Bob, and Carol sign messages to add Mallory as a signer, using `Signers.addSigner(...)` - (Tx#1)
2. After some discussion, they decide to remove Mallory from the signer list. They sign their messages and Alice (or anyone else) calls `Signers.removeSigner(...)`.
3. Mallory explores the blockchain and retrieves the transaction data from Tx#1 and calls `Signers.addSigner(...)` again, including himself on the signer list again.

Recommendation: We recommend adding a nonce and contract address to the signed message.

Update: Fixed on oasys-genesis-contract@96f3c3c, and oasys-genesis-contract@2edd5e2

QSP-9 Wrong Function Selector in Signature Verification

Severity: *High Risk*

Status: Fixed

File(s) affected: `contracts/nft-bridge/Signers.sol`

Description: In `Signers.removeSigner(...)`, the hash calculated for verifing the signature received uses `Signers.addSigner.selector`:

```
bytes32 _hash = keccak256(
    abi.encodeWithSelector(Signers.addSigner.selector, _address)
);
```

Recommendation: Compute hash for signature verification using `Signers.removeSigner.selector`.

Update: Fixed on oasys-genesis-contract@a6c0829.

QSP-10 Users May Not Be Able to Withdraw Funds in Early Stages

Severity: *High Risk*

Status: Fixed

File(s) affected: `packages/contracts/contracts/oasys/L1/L1BuildDeposit.sol`

Description: Users will not be able to withdraw funds from a non-built builder when the `block.number` is less than the configured `lockedBlock`.

Exploit Scenario:

1. User deposits OAS in a not-built Verse-Builder address using `deposit(...)`.
2. User tries to withdraw his/her funds using `withdraw(...)`:
   - As the builder is not built, `buildBlock[builder]` will be zero. Thus, for every `block.number` smaller than `lockedBlock`, the `require` statement in `L88` will be false.

```
require(buildBlock[_builder] + lockedBlock < block.number, "while OAS locked");
```

Recommendation: Add a check for handling the case where a builder is not already built. In that case, the user must be able to withdraw the funds.

Update: Fixed on oasys-optimism@24bb026.

QSP-11 Missing Specification

Severity: High Risk

Status: Acknowledged

Description: No specification was provided detailing the intended behavior of the audited smart contracts. Given that the audited smart contracts are high complexity, facilitating cross-chain bridging and building the Verse-Layer, the lack of documentation limited auditors' ability to detect mechanism design level issues.

Recommendation: Specify the intended use and users of each of the audited smart contracts.

Update: The Oasys team states that they will prepare further specifications of their smart contracts in the near future.

QSP-12 Validator Allowlist Introduces A Single Point Of Failure

Severity: Medium Risk

Status: Acknowledged

File(s) affected: `contracts/StakeManager.sol`, `contracts/lib/Allowlist.sol`

Description: Becoming an Oasys validator candidate requires: (i) allowlist addition, and (ii) having greater than or equal to `validatorThreshold` worth of tokens staked. The `Allowlist` contract inherits from OpenZeppelin's Ownable contract and allows the `owner` to add and remove addresses from the `allowlist`. Given that no documentation or code was provided explaining how the `Allowlist` will be run, we have assumed that it will be controlled by the Oasys team and as such is a single point of failure.

Recommendation: Reevaluate if requiring validators to be allowlisted is worth the additional point of failure. If it is not, remove the `Allowlist`. If it is, add both technical and end-user documentation explaining the following:

1. Who will control the `Allowlist`.
2. How modifications to the `Allowlist` will be made e.g., on-chain governance, off-chain governance, by the Oasys team, etc.
3. The risks and failure modes associated with requiring validators to be added to an `Allowlist`.

Update: The Oasys team states that the Oasys Foundation plans to dissolve in `6` years, at which time they will renounce ownership of the `Allowlist` and allow the `validator`s to participate freely. This policy is undocumented and will be documented as soon as possible.

QSP-13 Poor Inline Code Documentation

Severity: Medium Risk

Status: Acknowledged

File(s) affected: `All Smart Contracts`

Description: There was minimal inline code documentation which made it difficult for auditors to check that the code worked as intended.

Recommendation: Add inline code documentation to all smart contracts explaining the actions and the reasoning behind the actions that are being performed.

Update: The Oasys team states that they will improve their inline code documentation as soon as possible.

QSP-14 Allowlist Unbound Iteration Denial of Service

Severity: Medium Risk

Status: Fixed

File(s) affected: `contracts/lib/Allowlist.sol`

Description: `Allowlist`'s `_contains` function iterates through the `_allowlist`, comparing allowlisted `address`s with the given `address`, to determine whether or not a given address is allowlisted. The gas costs associated with iterating through the `_allowlist` increase linearly. In the worst case Allowlist's `_contains` function performs a `SSLOAD` read and its `removeAddress` function performs a read (`SSLOAD`) and a write (`SSTORE`) on each address in `_allowlist`. Given that the `_allowlist` has no upper bound on its size, the `Allowlist` owner could allowlist enough `address`es to make Allowlist's functions unusable due to their gas costs exceeding the maximum block gas limit.

Recommendation: Change `Allowlist`'s `_allowlist` to be a mapping instead of a list to make all `Allowlist` operations performable in constant instead of linear time.

Update: Fixed on oasys-genesis-contract@12310da.

QSP-15 Blocks Proposed By Unexpected Validators Accepted

Severity: Medium Risk

Status: Acknowledged

File(s) affected: `consensus/oasys/oasys.go`

Description: In `consensus/oasys/oasys.go` on `L657` to `L661` when `validator != expectedValidator` the function only returns an error if the program fails to `slash` the `validator`. In this case, the block generated by that `validator` is still going to be included in the blockchain. This is undesirable because this would lead to a fork of the blockchain.

Exploit Scenario: Besides slashing the `validator` who is an unexpected block producer, the block proposed by them should also be rejected. No state changes made based off of the invalidly proposed block should be persisted, except the changes made by calling `slash`.

Update: The Oasys team states that allowing any validator to propose a block if the `expectedValidator` does not is by design.

QSP-16 Predictable Schedule of Block Producers

Severity: Medium Risk

Status: Mitigated

File(s) affected: `consensus/oasys/oasys.go`

Description: According to the Oasys Technical Materials an epoch contains `5760` blocks where blocks are generated every `15` seconds. This means an epoch would last for `15 x 5760 = 86400` seconds (i.e., a day). An attacker knowing in the schedule for this long period of time in advance is dangerous since this information could be utilized to conduct attacks.

Also, `consensus/oasys/oasys.go`'s `getValidatorSchedule` function uses the block number at the end of each epoch as the seed for the random number generator. This allows an attacker to (i) impact validator selection by choosing the block number that will be used in the randomness seed and (ii) predict the schedule of all blocks in advance even further before each epoch.

Exploit Scenario: Here are some potential exploit scenarios:

- Exploit 1: Attacker could execute a Distributed-Denial-of-Service on some of the block producers in order to DoS the system, or to make sure the validators under their control become the block producer to perform a double-spending attack.
- Exploit 2: Malicious validators could team up and wait for a favorable schedule where conducting a double-spending attack is possible.
- Exploit 3: Could mix-up the attack vector mentioned in `Exploit1` and `Exploit2` and come up with a more efficient attack plan.

Recommendation: Change block proposer selection to be based off of randomness derived by multiple parties, for example see RandDAO. If multiple party randomness is deemed to be too complex or to have too high an implementation cost consider the following mitigations:

- Add the block hash of the final block of an epoch as the random seed. This would remove the ability of users to predict the block proposer `schedule` before entering an `epoch`.
- Reduce the blocks within an epoch from `5760` to a smaller number. For instance `20` (i.e., time length of an epoch would be reduced from `86400` seconds to `300` seconds). This would largely reduce the amount of time between when an attacker learns the block proposer `schedule` information and when they could use it to carry out an attack. This would increase the difficulty/complexity of an attack but the risk would still exist.

Update: The Oasys team modified `consensus/oasys/oasys.go`'s `newWeightedRandomChooser` to use the previous block's hash as a source of randomness. This makes block proposer schedule tampering more challenging. However, we still recommending reducing `epoch` times to be significantly smaller than `5760` to further decrease the likelihood of block proposer tampering. Commit: oasys-validator@01ecc77.

QSP-17 Signers Count Can Go Below Threshold

Severity: Medium Risk

Status: Fixed

File(s) affected: `contracts/nft-bridge/Signers.sol`

Description: The function `Signers.removeSigner(...)` does not check the threshold prior to removing a signer. Consequently, if, by mistake, `Signers.updateThreshold(...)` is not called before removing a signer, the contract may become useless since there won't be enough signers to sign a transaction. This will block all the operations that need signature verification, leaving all tokens controlled by it inaccessible.

Recommendation: Modify `removeSigner` to check that there will still be enough `signer`s to meet the `threshold`.

Update: Fixed on oasys-genesis-contract@091e635.

QSP-18 Signed Message May Be Replayed Across Chains

Severity: Medium Risk

Status: Fixed

File(s) affected: `contracts/nft-bridge/NFTBridgeRelayer.sol`, `contracts/nft-bridge/Signers.sol`

Related Issue(s): SWC-117, SWC-121, SWC-122

Description: Although the contract checks the validity of the signed message in `verifySignatures(...)`, the hashed value excludes a chain ID, and, therefore, the signed message may be reused across different Ethereum chains.

Recommendation: We recommend adding the chain ID (as retrieved from the underlying blockchain) to the signed message.

Update: Fixed on oasys-genesis-contract@0a08f40.

QSP-19 Missing Tests for NFT Bridge Contracts

Severity: Medium Risk

Status: Fixed

Description: The project does not implement tests for the new functionalities related to Oasys. Tests may express requirements and are necessary to validate the software's intent.

Recommendation: We highly recommend improving the test suite. Add tests for all of the audited smart contracts functionality. 100% code coverage, though not foolproof, should be targeted as it is a robust heuristic for gauging test thoroughness.

Update: Fixed on oasys-optimism@f805682, and oasys-genesis-contract@5d5c07a.

QSP-20 Unlocked Pragma

Severity: Low Risk

Status: Fixed

File(s) affected: `All Smart Contracts`

Related Issue(s): SWC-103

Description: Every Solidity file specifies in the header a version number of the format `pragma solidity (^)0.8.*`. The caret (^) before the version number implies an unlocked pragma, meaning that the compiler will use the specified version `0.8.0` and above, hence the term "unlocked".
All the Solidity files reviewed specify various Solidity pragma versions ranging from `0.4.2` to `0.8.9`. `WOAS` in particular may be compiled with Solidity compilers versions ranging from `0.4.22` < to `0.5.17`.

Recommendation: For consistency and to prevent unexpected behavior in the future, we recommend removing Solidity versioning containing carets and ranges. All Solidity contracts versioned `0.8.0` and greater should be locked at `0.8.12`. The `WOAS` contract should be locked at Solidity version `0.5.17` which is the latest compiler version that its compilation range allows.

Update: Fixed on

oasys-genesis-contract@c1ecda5, and oasys-optimism@1669f44.

QSP-21 Allowance Double-Spend Exploit

Severity: Low Risk

Status: Mitigated

File(s) affected: `contracts/token/WOAS.sol`

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

Exploit Scenario:

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

Recommendation: The exploit (as described above) is mitigated through use of functions that increase/decrease the allowance relative to its current value, such as `increaseAllowance()` and `decreaseAllowance()`. Furthermore, we recommend that developers of applications dependent on `approve()` / `transferFrom()` should keep in mind that they have to set allowance to 0 first and verify if it was before setting the new value.

Update: The Oasys team modified `WOAS` to inherit from OpenZeppelin's ERC20 contract which implements `increaseAllowance()` and `decreaseAllowance()`. Though the double allowance issue is still present, its presence is unavoidable in ERC20 compliant tokens. Commit: oasys-genesis-contract@6daaf50.

QSP-22 Signature Malleability

Severity: Low Risk

Status: Fixed

File(s) affected: `contracts/nft-bridge/Signers.sol`

Related Issue(s): SWC-117

Description: The given implementation of signature verification using `ecrecover` directly in `recoverSigner()` is prone to signature malleability.

Recommendation: Consider using a secure wrapper like OpenZeppelin's ECDSA utility library, which performs additional security checks on signature parameters.

Update: Fixed on oasys-genesis-contract@6182d0c.

QSP-23 Privileged Roles and Ownership

Severity: Low Risk

Status: Mitigated

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

Recommendation: This centralization of power needs to be made clear to the users, especially depending on the level of privilege the contract allows to the owner.

Update: Oasys team replied: *NFTBridgeMainchain and NFTBridgeSidechain are owned by NFTBridgeRelayer, so there is no problem.*

QSP-24 Signers May Be Duplicated

Severity: Low Risk

Status: Fixed

File(s) affected: `contracts/nft-bridge/Signers.sol`

Description: It is possible to initialize the contract in a way that there are multiple entries for the same address passed to the constructor.

Recommendation: Check `_signers` argument in `Signers.constructor(...)` for duplicates.

Update: Fixed on oasys-genesis-contract@ffa905a.

QSP-25 Ownership Can Be Renounced

Severity: Low Risk

Status: Fixed

File(s) affected: `contracts/nft-bridge/NFTBridgeMainchain.sol`, `contracts/nft-bridge/NFTBridgeSidechain.sol`

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

Recommendation: Double check if this is the intended behavior.

Update: Fixed on oasys-genesis-contract@6f82a79.

QSP-26 Ownership Can Be Transferred Bypassing Chain Verification

Severity: Low Risk

Status: Fixed

File(s) affected: `contracts/nft-bridge/NFTBridgeMainchain.sol`, `contracts/nft-bridge/NFTBridgeSidechain.sol`

Description: `NFTBridgeMainchain` and `NFTBridgeSidechain` contracts inherit from `Ownable` exposing `Ownable.transferOwnership(...)` with public visibility. An owner could call this function, bypassing the verification done in `NFTBridgeMainchain.transferMainchainRelayer(...)` and `NFTBridgeSidechain.transferSidechainRelayer(...)`:

```
L146 require(mainchainId == block.chainid, "Invalid main chain id.");
L147 super.transferOwnership(newRelayer);
```

```
L234 require(sidechainId == block.chainid, "Invalid side chain id");
L235 super.transferOwnership(newRelayer);
```

Recommendation: Override and disable `Ownable.transferOwnership(...)` to avoid this behavior.

Update: Fixed on oasys-genesis-contract@30e7e33.

QSP-27 Signatures Valid Indefinetly

Severity: Low Risk

Status: Fixed

File(s) affected: `contracts/nft-bridge/Signers.sol`

Description: `Signers.verifySignature(...)` function does not support signature expiration or renunciation. A malicious party could withhold `signer` signatures and then submit them at an inopportune time. Additionally, `signer`s are locked into what they have signed and cannot change their mind.

Recommendation: Add expiration to signed messages and have `verifySignature(...)` check that the signed message expiration has not passed.

Update: Fixed on oasys-genesis-contract@562eed7.

QSP-28 Unused Imports

Severity: Informational

Status: Fixed

File(s) affected: `contracts/token/SOAS.sol`

Description: `SOAS` imports OpenZeppelin's Ownable contract but does not make use of it.

Recommendation: We recommend evaluating whether Ownable is meant to be used. If it is, modify `SOAS` so use it. If it is not, then it should be removed to increase code readability.

Update: Fixed on oasys-genesis-contract@5fcd060.

QSP-29 Ownership Can Be Renounced

Severity: Informational

Status: Acknowledged

File(s) affected: `contracts/Allowlist.sol`

Description: If the `owner` renounces their ownership, all `Ownable` contracts will be left without an owner. Consequently, any function guarded by the `onlyOwner` modifier will no longer be able to be executable.
`Allowlist` inherits from OpenZeppelin's `Ownable` contract which includes ownership renunciation functionality. If ownership functionality is renounced, addresses will no longer be able to be added to or removed from the `Allowlist`.

Recommendation: Double check if this is the intended behavior. If it is not, consider overriding and removing Ownable's `renounceOwnership` functionality. If it is, add end-user documentation explaining how ownership renunciation will be used.

Update: The Oasys team states that the Oasys Foundation plans to dissolve in 6 years, at which time they will renounce ownership of the `Allowlist`, to support this, ownership renunciation must be left in `Allowlist`.

QSP-30 Allowance Double-Spend Exploit

Severity: Informational

Status: Acknowledged

File(s) affected: `contracts/token/SOAS.sol`

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

Exploit Scenario:

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

5. Before Alice notices any irregularities, Bob calls `transferFrom()` method again, this time to transfer `M` Alice's tokens.

Recommendation: The exploit (as described above) is mitigated through use of functions that increase/decrease the allowance relative to its current value, such as `increaseAllowance()` and `decreaseAllowance()`. Furthermore, we recommend that developers of applications dependent on `approve()` / `transferFrom()` should keep in mind that they have to set allowance to 0 first and verify if it was before setting the new value.

Update: The Oasys team states that contract users will be informed to use `increaseAllowance()`and `decreaseAllowance()`.

QSP-31 Transaction Ordering Dependence for `initialize()` Functions

Severity: Informational

Status: Acknowledged

File(s) affected: `packages/contracts/contracts/oasys/L1/build/L1BuildDeposit.sol`

Related Issue(s): SWC-114

Description: The various `initialize()` functions of contracts are not constructors. There's a low-but-not-zero chance that someone can call these after the contracts have been deployed but before the development team calls them. Consequently, contracts may get initialized with values that are not desirable by the development team.

Recommendation: Be aware of this issue, and be prepared to redeploy your contracts if these calls are front-run. Do not use your project until you have checked that your calls to these functions went through. Another possible mitigation is to add access control to `initialize(...)` so that only a privileged user can call it.

QSP-32 Calculations Based on Current Environment Value Parameters

Severity: Undetermined

Status: Fixed

File(s) affected: `consensus/oasys/contract.go`, `contracts/StakeManager.sol`

Description: `consensus/oasys/contract.go` `IsEpoch` and `Epoch` functions take in a block number and evaluate it based off of the current `environmentValue`. Similarly, multiple functions in the `StakeManager` contract use `block.number` and evaluate it based on the current `EnvironmentValue`. However, none of the specified functionality takes into account previous `EnvironmentValue`s with different parameters. This could lead to situations where incorrect information is returned to function callers.

Recommendation: Modify all functions using `EnvironmentValue`s in both the Oasys Validator Client and the `StakeManager` contract to take into account previous `EnvironmentValue`s.

Update: Response from the Oasys team: *There is no problem because the past* `EnvironmentValue` *is stored in snapshot*.

QSP-33 ERC721 Token May Be Blocked when Finalizing Withdrawal

Severity: Undetermined

Status: Fixed

File(s) affected: `contracts/nft-bridge/NFTBridgeMainchain.sol`

Description: In `NFTBridgeMainchain.finalizeWithdrawal(...)` a try-catch statement is used to manage possible errors raised in `IERC721(mainInfo.mainchainERC721).safeTransferFrom(...)`. If this call fails, the error will be caught and the catch block will be executed (emitting `WithdrawalFailed` event). However, the transaction will be finished successfully, without reverting. This means that the storage modification done in line 109 will be committed:

```
L109 mainInfo.mainTo = mainTo;
```

If the owner tries to finalize the withdrawal again, it will revert as this field is not zero anymore:

```
L107 require(mainInfo.mainTo == address(0), "already withdraw");
```

Recommendation: Confirm this is the intended behavior. Otherwise, revert the storage modification or revert the transaction.

Update: Fixed on oasys-genesis-contract@1bf1633.

QSP-34 ERC721 Token May Be Blocked when Finalizing Deposit

Severity: Undetermined

Status: Fixed

File(s) affected: `contracts/nft-bridge/NFTBridgeSidechain.sol`

Description: In `NFTBridgeSidechain.finalizeDeposit(...)` a try-catch statement is used to manage possible errors raised in `SidechainERC721(sidechainERC721).mint(...)`.
If this call fails, the error will be caught and the catch block will be executed (emitting `DepositeFailed` event).
However, the transaction will be finished successfully, without reverting. This means that the storage modification done in line 129 will be committed:

```
L129 depositIndexes[mainchainId][depositIndex] = true;
```

If the owner tries to finalize the deposit again, it will revert as this field is not zero anymore:

```
L125 require(
L126     !depositIndexes[mainchainId][depositIndex],
L127     "Already deposited"
L128 );
```

Recommendation: Confirm this is the intended behavior. Otherwise, revert the storage modification or revert the transaction.

Update: Fixed on oasys-genesis-contract@f8ebdff.

Automated Analyses

Slither

All the issues found by Slither were discussed in the sections of this report. Please note that most of them were triaged as false positives and/or related to files out of scope.

oasys-genesis-contract

```
Reentrancy in SOAS.claim(uint256) (contracts/token/SOAS.sol#80-91):
    External calls:
    - (success) = msg.sender.call{value: amount}(new bytes(0)) (contracts/token/SOAS.sol#85)
    State variables written after the call(s):
    - claimInfo[msg.sender].claimed += amount (contracts/token/SOAS.sol#88)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities

Validator.getRewards(IStakeManager.Validator,IEnvironment.EnvironmentValue,uint256) (contracts/lib/Validator.sol#121-145) performs a multiplication on the result of a division:
    -rewards = (_stake * Math.percent(env.rewardRate,Constants.MAX_REWARD_RATE,Constants.REWARD_PRECISION)) / 10 ** Constants.REWARD_PRECISION (contracts/lib/Validator.sol#133-135)
    -rewards *= Math.percent(env.blockPeriod * env.epochPeriod,Constants.SECONDS_PER_YEAR,Constants.REWARD_PRECISION) (contracts/lib/Validator.sol#138-142)
BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#91-226) performs a multiplication on the result of a division:
    -sstore(uint256,uint256)(_preBytes,fslot_concatStorage_asm_0 + mload(uint256)(_postBytes + 0x20) / 0x100 ** 32 - mlength_concatStorage_asm_0 * 0x100 ** 32 - newlength_concatStorage_asm_0 + mlength_concatStorage_asm_0 *
2) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#115-140)
BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#91-226) performs a multiplication on the result of a division:
    -sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#189)
BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#91-226) performs a multiplication on the result of a division:
    -sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#223)
BytesLib.equalStorage(bytes,bytes) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#439-509) performs a multiplication on the result of a division:
    -fslot_equalStorage_asm_0 = fslot_equalStorage_asm_0 / 0x100 * 0x100 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#466)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply

Environment.isFirstBlock() (contracts/Environment.sol#69-71) uses a dangerous strict equality:
    - (block.number) % value().epochPeriod == 0 (contracts/Environment.sol#70)
Environment.isLastBlock() (contracts/Environment.sol#76-78) uses a dangerous strict equality:
    - (block.number + 1) % value().epochPeriod == 0 (contracts/Environment.sol#77)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Contract locking ether found:
    Contract TestERC20 (contracts/test/TestERC20.sol#7-14) has payable functions:
     - TestERC20.mint() (contracts/test/TestERC20.sol#11-13)
    But does not have a function to withdraw the ether
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#contracts-that-lock-ether

Reentrancy in NFTBridgeSidechain.finalizeDeposit(uint256,uint256,uint256,address,uint256,address,address) (contracts/nft-bridge/NFTBridgeSidechain.sol#97-154):
    External calls:
    - SidechainERC721(sidechainERC721).mint(sideTo,tokenId) (contracts/nft-bridge/NFTBridgeSidechain.sol#130-153)
    State variables written after the call(s):
    - _depositIndexes[mainchainId][depositIndex] = true (contracts/nft-bridge/NFTBridgeSidechain.sol#131)
Reentrancy in NFTBridgeMainchain.finalizeWithdrawal(uint256,uint256,uint256,uint256,address,address) (contracts/nft-bridge/NFTBridgeMainchain.sol#98-139):
    External calls:
    - IERC721(mainInfo.mainchainERC721).safeTransferFrom(address(this),mainTo,mainInfo.tokenId) (contracts/nft-bridge/NFTBridgeMainchain.sol#112-138)
    State variables written after the call(s):
    - mainInfo.mainTo = mainTo (contracts/nft-bridge/NFTBridgeMainchain.sol#119)
Reentrancy in NFTBridgeRelayer.transferMainchainRelayer(uint256,address,uint64,bytes) (contracts/nft-bridge/NFTBridgeRelayer.sol#100-128):
    External calls:
    - INFTBridgeMainchain(mainchainBridge).transferMainchainRelayer(mainchainId,newRelayer) (contracts/nft-bridge/NFTBridgeRelayer.sol#122-125)
    State variables written after the call(s):
    - nonce ++ (contracts/nft-bridge/NFTBridgeRelayer.sol#127)
Reentrancy in NFTBridgeRelayer.transferSidechainRelayer(uint256,address,uint64,bytes) (contracts/nft-bridge/NFTBridgeRelayer.sol#226-254):
    External calls:
    - INFTBridgeSidechain(sidechainBridge).transferSidechainRelayer(sidechainId,newRelayer) (contracts/nft-bridge/NFTBridgeRelayer.sol#248-251)
    State variables written after the call(s):
    - nonce ++ (contracts/nft-bridge/NFTBridgeRelayer.sol#253)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

Staker.unstake(IStakeManager.Staker,IEnvironment,IStakeManager.Validator,Token.Type,uint256).refunds (contracts/lib/Staker.sol#59) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

ERC721._checkOnERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#388-409) ignores return value by
IERC721Receiver(to).onERC721Received(_msgSender(),from,tokenId,_data) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#395-405)
Validator.unstake(IStakeManager.Validator,IEnvironment,uint256) (contracts/lib/Validator.sol#67-73) ignores return value by validator.stakeUpdates.sub(validator.stakeAmounts,environment.epoch() + 1,amount)
(contracts/lib/Validator.sol#72)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

NFTBridgeRelayer.constructor(address,address,address[],uint256).threshold (contracts/nft-bridge/NFTBridgeRelayer.sol#30) shadows:
    - Signers.threshold (contracts/nft-bridge/Signers.sol#21) (state variable)
SidechainERC721.constructor(uint256,address,string,string)._name (contracts/nft-bridge/SidechainERC721.sol#35) shadows:
    - ERC721._name (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#24) (state variable)
SidechainERC721.constructor(uint256,address,string,string)._symbol (contracts/nft-bridge/SidechainERC721.sol#36) shadows:
    - ERC721._symbol (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#27) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

NFTBridgeRelayer.constructor(address,address,address[],uint256)._mainchainBridge (contracts/nft-bridge/NFTBridgeRelayer.sol#27) lacks a zero-check on :
        - mainchainBridge = _mainchainBridge (contracts/nft-bridge/NFTBridgeRelayer.sol#32)
NFTBridgeRelayer.constructor(address,address,address[],uint256)._sidechainBridge (contracts/nft-bridge/NFTBridgeRelayer.sol#28) lacks a zero-check on :
        - sidechainBridge = _sidechainBridge (contracts/nft-bridge/NFTBridgeRelayer.sol#33)
SidechainERC721.constructor(uint256,address,string,string).mainchainERC721 (contracts/nft-bridge/SidechainERC721.sol#34) lacks a zero-check on :
        - _mainchainERC721 = mainchainERC721 (contracts/nft-bridge/SidechainERC721.sol#39)
SOAS.constructor(address)._staking (contracts/token/SOAS.sol#47) lacks a zero-check on :
        - staking = _staking (contracts/token/SOAS.sol#48)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).retval (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#395)' in ERC721._checkOnERC721Received(address,address,uint256,bytes)
(node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#388-409) potentially used before declaration: retval == IERC721Receiver.onERC721Received.selector
(node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#396)
Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).reason (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#397)' in ERC721._checkOnERC721Received(address,address,uint256,bytes)
(node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#388-409) potentially used before declaration: reason.length == 0 (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#398)
Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).reason (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#397)' in ERC721._checkOnERC721Received(address,address,uint256,bytes)
(node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#388-409) potentially used before declaration: revert(uint256,uint256)(32 + reason,mload(uint256)(reason))
(node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#402)
Variable 'ECDSA.tryRecover(bytes32,bytes).r (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#62)' in ECDSA.tryRecover(bytes32,bytes) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#57-86)
potentially used before declaration: r = mload(uint256)(signature + 0x20) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#79)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sc_concatStorage_asm_0 = keccak256(uint256,uint256)(0x0,0x20) + slength_concatStorage_asm_0 / 32 (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#195)
Variable 'BytesLib.concatStorage(bytes,bytes).submod_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#161)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: submod_concatStorage_asm_0 = 32 - slengthmod_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#204)
Variable 'BytesLib.concatStorage(bytes,bytes).submod_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#161)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mc_concatStorage_asm_0 = _postBytes + submod_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#205)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mc_concatStorage_asm_0 = _postBytes + submod_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#205)
Variable 'BytesLib.concatStorage(bytes,bytes).end_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#163)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: end_concatStorage_asm_0 = _postBytes + mlength_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#206)
Variable 'BytesLib.concatStorage(bytes,bytes).mask_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#164)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mask_concatStorage_asm_0 = 0x100 ** submod_concatStorage_asm_0 - 1 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#207)
Variable 'BytesLib.concatStorage(bytes,bytes).submod_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#161)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mask_concatStorage_asm_0 = 0x100 ** submod_concatStorage_asm_0 - 1 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#207)
Variable 'BytesLib.concatStorage(bytes,bytes).mask_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#164)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sstore(uint256,uint256)(sc_concatStorage_asm_0,sload(uint256)(sc_concatStorage_asm_0) + mload(uint256)(mc_concatStorage_asm_0) &
mask_concatStorage_asm_0) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#209)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sstore(uint256,uint256)(sc_concatStorage_asm_0,sload(uint256)(sc_concatStorage_asm_0) + mload(uint256)(mc_concatStorage_asm_0) &
mask_concatStorage_asm_0) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#209)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sstore(uint256,uint256)(sc_concatStorage_asm_0,sload(uint256)(sc_concatStorage_asm_0) + mload(uint256)(mc_concatStorage_asm_0) &
mask_concatStorage_asm_0) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#209)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sc_concatStorage_asm_0 = sc_concatStorage_asm_0 + 1 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#212)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mc_concatStorage_asm_0 = mc_concatStorage_asm_0 + 0x20 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#213)
Variable 'BytesLib.concatStorage(bytes,bytes).end_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#163)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mc_concatStorage_asm_0 < end_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#214)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mc_concatStorage_asm_0 < end_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#214)
Variable 'BytesLib.concatStorage(bytes,bytes).mask_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#164)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mask_concatStorage_asm_0 = 0x100 ** mc_concatStorage_asm_0 - end_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#221)
Variable 'BytesLib.concatStorage(bytes,bytes).end_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#163)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mask_concatStorage_asm_0 = 0x100 ** mc_concatStorage_asm_0 - end_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#221)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mask_concatStorage_asm_0 = 0x100 ** mc_concatStorage_asm_0 - end_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#221)
Variable 'BytesLib.concatStorage(bytes,bytes).mask_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#164)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0)
```

```
(node_modules/solidity-bytes-utils/contracts/BytesLib.sol#223)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0)
(node_modules/solidity-bytes-utils/contracts/BytesLib.sol#223)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0)
(node_modules/solidity-bytes-utils/contracts/BytesLib.sol#223)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0)) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#218)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0)) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#218)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: sc_concatStorage_asm_0 = sc_concatStorage_asm_0 + 1 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#215)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-
utils/contracts/BytesLib.sol#91-226) potentially used before declaration: mc_concatStorage_asm_0 = mc_concatStorage_asm_0 + 0x20 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#216)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables

Reentrancy in NFTBridgeMainchain.deposit(address,uint256,uint256,address) (contracts/nft-bridge/NFTBridgeMainchain.sol#38-63):
    External calls:
    - IERC721(mainchainERC721).transferFrom(msg.sender,address(this),tokenId) (contracts/nft-bridge/NFTBridgeMainchain.sol#46-50)
    State variables written after the call(s):
    - _depositInfos.push(DepositInfo(mainchainERC721,tokenId,msg.sender,address(0))) (contracts/nft-bridge/NFTBridgeMainchain.sol#51-53)
Reentrancy in NFTBridgeSidechain.finalizeDeposit(uint256,uint256,uint256,address,uint256,address,address) (contracts/nft-bridge/NFTBridgeSidechain.sol#97-154):
    External calls:
    - SidechainERC721(sidechainERC721).mint(sideTo,tokenId) (contracts/nft-bridge/NFTBridgeSidechain.sol#130-153)
    State variables written after the call(s):
    - _depositIndexMap[sidechainERC721][tokenId] = depositIndex (contracts/nft-bridge/NFTBridgeSidechain.sol#132)
Reentrancy in StakeManager.stake(address,Token.Type,uint256) (contracts/StakeManager.sol#259-274):
    External calls:
    - Token.receives(token,msg.sender,amount) (contracts/StakeManager.sol#266)
    State variables written after the call(s):
    - stakerSigners.push(msg.sender) (contracts/StakeManager.sol#270)
    - staker.signer = msg.sender (contracts/StakeManager.sol#269)
Reentrancy in NFTBridgeSidechain.withdraw(address,uint256,address) (contracts/nft-bridge/NFTBridgeSidechain.sol#162-190):
    External calls:
    - SidechainERC721(sidechainERC721).burn(msg.sender,tokenId) (contracts/nft-bridge/NFTBridgeSidechain.sol#175)
    State variables written after the call(s):
    - _withdrawalInfos.push(WithdrawalInfo(sidechainERC721,tokenId,msg.sender,false)) (contracts/nft-bridge/NFTBridgeSidechain.sol#176-178)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in SOAS.claim(uint256) (contracts/token/SOAS.sol#80-91):
    External calls:
    - (success) = msg.sender.call{value: amount}(new bytes(0)) (contracts/token/SOAS.sol#85)
    Event emitted after the call(s):
    - Claim(msg.sender,amount) (contracts/token/SOAS.sol#90)
Reentrancy in NFTBridgeMainchain.deposit(address,uint256,uint256,address) (contracts/nft-bridge/NFTBridgeMainchain.sol#38-63):
    External calls:
    - IERC721(mainchainERC721).transferFrom(msg.sender,address(this),tokenId) (contracts/nft-bridge/NFTBridgeMainchain.sol#46-50)
    Event emitted after the call(s):
    - DepositeInitiated(_depositInfos.length - 1,mainchainERC721,tokenId,sidechainId,msg.sender,sideTo) (contracts/nft-bridge/NFTBridgeMainchain.sol#55-62)
Reentrancy in NFTBridgeSidechain.finalizeDeposit(uint256,uint256,uint256,address,uint256,address,address) (contracts/nft-bridge/NFTBridgeSidechain.sol#97-154):
    External calls:
    - SidechainERC721(sidechainERC721).mint(sideTo,tokenId) (contracts/nft-bridge/NFTBridgeSidechain.sol#130-153)
    Event emitted after the call(s):
    - DepositeFailed(mainchainId,depositIndex,mainchainERC721,sidechainERC721,tokenId,mainFrom,sideTo) (contracts/nft-bridge/NFTBridgeSidechain.sol#144-152)
    - DepositeFinalized(mainchainId,depositIndex,mainchainERC721,sidechainERC721,tokenId,mainFrom,sideTo) (contracts/nft-bridge/NFTBridgeSidechain.sol#134-142)
Reentrancy in NFTBridgeMainchain.finalizeWithdrawal(uint256,uint256,uint256,uint256,address,address) (contracts/nft-bridge/NFTBridgeMainchain.sol#98-139):
    External calls:
    - IERC721(mainInfo.mainchainERC721).safeTransferFrom(address(this),mainTo,mainInfo.tokenId) (contracts/nft-bridge/NFTBridgeMainchain.sol#112-138)
    Event emitted after the call(s):
    - WithdrawalFailed(depositIndex,sidechainId,withdrawalIndex,mainInfo.mainchainERC721,sideFrom,mainTo) (contracts/nft-bridge/NFTBridgeMainchain.sol#130-137)
    - WithdrawalFinalized(depositIndex,sidechainId,withdrawalIndex,mainInfo.mainchainERC721,sideFrom,mainTo) (contracts/nft-bridge/NFTBridgeMainchain.sol#121-128)
Reentrancy in NFTBridgeMainchain.rejectDeposit(uint256,uint256) (contracts/nft-bridge/NFTBridgeMainchain.sol#70-87):
    External calls:
    - IERC721(mainInfo.mainchainERC721).transferFrom(address(this),mainInfo.mainTo,mainInfo.tokenId) (contracts/nft-bridge/NFTBridgeMainchain.sol#80-84)
    Event emitted after the call(s):
    - DepositeRejected(depositIndex) (contracts/nft-bridge/NFTBridgeMainchain.sol#86)
Reentrancy in NFTBridgeSidechain.rejectWithdrawal(uint256,uint256) (contracts/nft-bridge/NFTBridgeSidechain.sol#197-224):
    External calls:
    - SidechainERC721(sideWithdrawal.sidechainERC721).mint(sideWithdrawal.sideFrom,sideWithdrawal.tokenId) (contracts/nft-bridge/NFTBridgeSidechain.sol#209-212)
    Event emitted after the call(s):
    - WithdrawalRejected(withdrawalIndex,mainchainId,_depositIndexMap[sideWithdrawal.sidechainERC721][sideWithdrawal.tokenId]) (contracts/nft-bridge/NFTBridgeSidechain.sol#217-223)
Reentrancy in SOAS.renounce(uint256) (contracts/token/SOAS.sol#97-106):
    External calls:
    - (success) = info.from.call{value: amount}(new bytes(0)) (contracts/token/SOAS.sol#102)
    Event emitted after the call(s):
    - Renounce(msg.sender,amount) (contracts/token/SOAS.sol#105)
Reentrancy in StakeManager.stake(address,Token.Type,uint256) (contracts/StakeManager.sol#259-274):
    External calls:
    - Token.receives(token,msg.sender,amount) (contracts/StakeManager.sol#266)
    Event emitted after the call(s):
    - Staked(msg.sender,validator,token,amount) (contracts/StakeManager.sol#273)
Reentrancy in StakeManager.unstake(address,Token.Type,uint256) (contracts/StakeManager.sol#279-288):
    External calls:
    - amount = stakers[msg.sender].unstake(environment,validators[validator],token,amount) (contracts/StakeManager.sol#286)
    Event emitted after the call(s):
    - Unstaked(msg.sender,validator,token,amount) (contracts/StakeManager.sol#287)
Reentrancy in NFTBridgeSidechain.withdraw(address,uint256,address) (contracts/nft-bridge/NFTBridgeSidechain.sol#162-190):
    External calls:
    - SidechainERC721(sidechainERC721).burn(msg.sender,tokenId) (contracts/nft-bridge/NFTBridgeSidechain.sol#175)
    Event emitted after the call(s):
    - WithdrawalInitiated(_withdrawalInfos.length - 1,mainchainId,_depositIndexMap[sidechainERC721][tokenId],mainchainERC721,sidechainERC721,tokenId,msg.sender,mainTo) (contracts/nft-bridge/NFTBridgeSidechain.sol#180-189)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Signers.verifySignatures(bytes32,uint64,bytes) (contracts/nft-bridge/Signers.sol#52-82) uses timestamp for comparisons
    Dangerous comparisons:
    - require(bool,string)(expiration >= block.timestamp,Signature expired) (contracts/nft-bridge/Signers.sol#58)
SOAS.mint(address,uint256,uint256) (contracts/token/SOAS.sol#61-74) uses timestamp for comparisons
    Dangerous comparisons:
    - require(bool,string)(block.timestamp < since && since < until,invalid since or until) (contracts/token/SOAS.sol#67)
SOAS.claim(uint256) (contracts/token/SOAS.sol#80-91) uses timestamp for comparisons
    Dangerous comparisons:
    - require(bool,string)(currentClaimableOAS >= amount,over claimable OAS) (contracts/token/SOAS.sol#82)
SOAS.getClaimableOAS(address) (contracts/token/SOAS.sol#120-133) uses timestamp for comparisons
    Dangerous comparisons:
    - block.timestamp < info.since (contracts/token/SOAS.sol#125)
    - amount > info.amount (contracts/token/SOAS.sol#129)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

ERC721._checkOnERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#388-409) uses assembly
    - INLINE ASM (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#401-403)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#201-221) uses assembly
    - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#213-216)
ECDSA.tryRecover(bytes32,bytes) (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#57-86) uses assembly
    - INLINE ASM (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#67-71)
    - INLINE ASM (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#78-81)
BytesLib.concat(bytes,bytes) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#13-89) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#23-86)
BytesLib.concatStorage(bytes,bytes) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#91-226) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#92-225)
BytesLib.slice(bytes,uint256,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#228-295) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#242-292)
BytesLib.toAddress(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#297-306) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#301-303)
BytesLib.toUint8(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#308-317) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#312-314)
BytesLib.toUint16(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#319-328) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#323-325)
BytesLib.toUint32(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#330-339) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#334-336)
BytesLib.toUint64(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#341-350) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#345-347)
BytesLib.toUint96(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#352-361) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#356-358)
BytesLib.toUint128(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#363-372) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#367-369)
BytesLib.toUint256(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#374-383) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#378-380)
BytesLib.toBytes32(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#385-394) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#389-391)
BytesLib.equal(bytes,bytes) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#396-437) uses assembly
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#399-434)
BytesLib.equalStorage(bytes,bytes) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#439-509) uses assembly
```

```
    - INLINE ASM (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#449-506)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
    - Version used: ['0.8.12', '>=0.8.0<0.9.0', '^0.8.0', '^0.8.1', '^0.8.9']
    - ^0.8.0 (node_modules/@openzeppelin/contracts/access/Ownable.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/IERC721Enumerable.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/IERC721Metadata.sol#4)
    - ^0.8.1 (node_modules/@openzeppelin/contracts/utils/Address.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#4)
    - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#4)
    - ^0.8.0 (contracts/Environment.sol#3)
    - ^0.8.0 (contracts/IEnvironment.sol#3)
    - ^0.8.0 (contracts/IStakeManager.sol#3)
    - ^0.8.0 (contracts/StakeManager.sol#3)
    - ^0.8.0 (contracts/System.sol#3)
    - ^0.8.0 (contracts/lib/Allowlist.sol#3)
    - ^0.8.0 (contracts/lib/Constants.sol#3)
    - ^0.8.0 (contracts/lib/EnvironmentValue.sol#3)
    - ^0.8.0 (contracts/lib/IAllowlist.sol#3)
    - ^0.8.0 (contracts/lib/Math.sol#3)
    - ^0.8.0 (contracts/lib/Staker.sol#3)
    - ^0.8.0 (contracts/lib/Token.sol#3)
    - ^0.8.0 (contracts/lib/UpdateHistories.sol#3)
    - ^0.8.0 (contracts/lib/Validator.sol#3)
    - 0.8.12 (contracts/nft-bridge/INFTBridgeMainchain.sol#2)
    - 0.8.12 (contracts/nft-bridge/INFTBridgeSidechain.sol#2)
    - 0.8.12 (contracts/nft-bridge/NFTBridgeMainchain.sol#2)
    - 0.8.12 (contracts/nft-bridge/NFTBridgeRelayer.sol#2)
    - 0.8.12 (contracts/nft-bridge/NFTBridgeSidechain.sol#2)
    - 0.8.12 (contracts/nft-bridge/SidechainERC721.sol#2)
    - 0.8.12 (contracts/nft-bridge/Signers.sol#2)
    - ^0.8.0 (contracts/test/TestERC20.sol#3)
    - ^0.8.9 (contracts/test/TestERC721.sol#3)
    - ^0.8.9 (contracts/token/SOAS.sol#2)
    - >=0.8.0<0.9.0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#9)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Pragma version>=0.4.22<0.6 (contracts/token/WOAS.sol#16) is known to contain severe issues (https://solidity.readthedocs.io/en/latest/bugs.html)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/Ownable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/IERC721Enumerable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/IERC721Metadata.sol#4) allows old versions
Pragma version^0.8.1 (node_modules/@openzeppelin/contracts/utils/Address.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/cryptography/ECDSA.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#4) allows old versions
Pragma version^0.8.0 (contracts/Environment.sol#3) allows old versions
Pragma version^0.8.0 (contracts/IEnvironment.sol#3) allows old versions
Pragma version^0.8.0 (contracts/IStakeManager.sol#3) allows old versions
Pragma version^0.8.0 (contracts/StakeManager.sol#3) allows old versions
Pragma version^0.8.0 (contracts/System.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/Allowlist.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/Constants.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/EnvironmentValue.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/IAllowlist.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/Math.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/Staker.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/Token.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/UpdateHistories.sol#3) allows old versions
Pragma version^0.8.0 (contracts/lib/Validator.sol#3) allows old versions
Pragma version0.8.12 (contracts/nft-bridge/INFTBridgeMainchain.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.12 (contracts/nft-bridge/INFTBridgeSidechain.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.12 (contracts/nft-bridge/NFTBridgeMainchain.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.12 (contracts/nft-bridge/NFTBridgeRelayer.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.12 (contracts/nft-bridge/NFTBridgeSidechain.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.12 (contracts/nft-bridge/SidechainERC721.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.12 (contracts/nft-bridge/Signers.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.0 (contracts/test/TestERC20.sol#3) allows old versions
Pragma version^0.8.9 (contracts/test/TestERC721.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version^0.8.9 (contracts/token/SOAS.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version>=0.8.0<0.9.0 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#9) is too complex
solc-0.8.12 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#60-65):
    - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts/utils/Address.sol#63)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#128-139):
    - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#137)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#157-166):
    - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#164)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#184-193):
    - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#191)
Low level call in Token.transfers(Token.Type,address,uint256) (contracts/lib/Token.sol#51-63):
    - (success) = to.call{value: amount}(new bytes(0)) (contracts/lib/Token.sol#58)
Low level call in SOAS.claim(uint256) (contracts/token/SOAS.sol#80-91):
    - (success) = msg.sender.call{value: amount}(new bytes(0)) (contracts/token/SOAS.sol#85)
Low level call in SOAS.renounce(uint256) (contracts/token/SOAS.sol#97-106):
    - (success) = info.from.call{value: amount}(new bytes(0)) (contracts/token/SOAS.sol#102)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter ERC721.safeTransferFrom(address,address,uint256,bytes)._data (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#179) is not in mixedCase
Parameter StakeManager.initialize(IEnvironment,IAllowlist)._environment (contracts/StakeManager.sol#115) is not in mixedCase
Parameter StakeManager.initialize(IEnvironment,IAllowlist)._allowlist (contracts/StakeManager.sol#115) is not in mixedCase
Parameter Allowlist.addAddress(address)._address (contracts/lib/Allowlist.sol#28) is not in mixedCase
Parameter Allowlist.removeAddress(address)._address (contracts/lib/Allowlist.sol#39) is not in mixedCase
Parameter Allowlist.containsAddress(address)._address (contracts/lib/Allowlist.sol#67) is not in mixedCase
Parameter EnvironmentValue.started(IEnvironment.EnvironmentValue,uint256)._block (contracts/lib/EnvironmentValue.sol#24) is not in mixedCase
Parameter Staker.getStakes(IStakeManager.Staker,address[],Token.Type,uint256)._validators (contracts/lib/Staker.sol#99) is not in mixedCase
Parameter Signers.verifySignatures(bytes32,uint64,bytes)._hash (contracts/nft-bridge/Signers.sol#53) is not in mixedCase
Parameter Signers.addSigner(address,uint64,bytes)._address (contracts/nft-bridge/Signers.sol#113) is not in mixedCase
Parameter Signers.removeSigner(address,uint64,bytes)._address (contracts/nft-bridge/Signers.sol#144) is not in mixedCase
Parameter Signers.updateThreshold(uint256,uint64,bytes)._threshold (contracts/nft-bridge/Signers.sol#181) is not in mixedCase
Parameter BytesLib.concat(bytes,bytes)._preBytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#14) is not in mixedCase
Parameter BytesLib.concat(bytes,bytes)._postBytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#15) is not in mixedCase
Parameter BytesLib.concatStorage(bytes,bytes)._preBytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#91) is not in mixedCase
Parameter BytesLib.concatStorage(bytes,bytes)._postBytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#91) is not in mixedCase
Parameter BytesLib.slice(bytes,uint256,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#229) is not in mixedCase
Parameter BytesLib.slice(bytes,uint256,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#230) is not in mixedCase
Parameter BytesLib.slice(bytes,uint256,uint256)._length (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#231) is not in mixedCase
Parameter BytesLib.toAddress(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#297) is not in mixedCase
Parameter BytesLib.toAddress(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#297) is not in mixedCase
Parameter BytesLib.toUint8(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#308) is not in mixedCase
Parameter BytesLib.toUint8(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#308) is not in mixedCase
Parameter BytesLib.toUint16(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#319) is not in mixedCase
Parameter BytesLib.toUint16(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#319) is not in mixedCase
Parameter BytesLib.toUint32(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#330) is not in mixedCase
Parameter BytesLib.toUint32(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#330) is not in mixedCase
Parameter BytesLib.toUint64(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#341) is not in mixedCase
Parameter BytesLib.toUint64(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#341) is not in mixedCase
Parameter BytesLib.toUint96(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#352) is not in mixedCase
Parameter BytesLib.toUint96(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#352) is not in mixedCase
Parameter BytesLib.toUint128(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#363) is not in mixedCase
Parameter BytesLib.toUint128(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#363) is not in mixedCase
Parameter BytesLib.toUint256(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#374) is not in mixedCase
Parameter BytesLib.toUint256(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#374) is not in mixedCase
Parameter BytesLib.toBytes32(bytes,uint256)._bytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#385) is not in mixedCase
Parameter BytesLib.toBytes32(bytes,uint256)._start (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#385) is not in mixedCase
Parameter BytesLib.equal(bytes,bytes)._preBytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#396) is not in mixedCase
Parameter BytesLib.equal(bytes,bytes)._postBytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#396) is not in mixedCase
Parameter BytesLib.equalStorage(bytes,bytes)._preBytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#440) is not in mixedCase
Parameter BytesLib.equalStorage(bytes,bytes)._postBytes (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#441) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Reentrancy in WOAS.withdraw(uint256) (contracts/token/WOAS.sol#38-43):
    External calls:
    - msg.sender.transfer(wad) (contracts/token/WOAS.sol#41)
    Event emitted after the call(s):
    - Withdrawal(msg.sender,wad) (contracts/token/WOAS.sol#42)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

Token.slitherConstructorConstantVariables() (contracts/lib/Token.sol#7-73) uses literals with too many digits:
    - WOAS = IERC20(0x5200000000000000000000000000000000000001) (contracts/lib/Token.sol#22)
Token.slitherConstructorConstantVariables() (contracts/lib/Token.sol#7-73) uses literals with too many digits:
    - SOAS = IERC20(0x5200000000000000000000000000000000000002) (contracts/lib/Token.sol#24)
BytesLib.toAddress(bytes,uint256) (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#297-306) uses literals with too many digits:
    - tempAddress = mload(uint256)(_bytes + 0x20 + _start) / 0x1000000000000000000000000 (node_modules/solidity-bytes-utils/contracts/BytesLib.sol#302)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

WOAS.decimals (contracts/token/WOAS.sol#21) should be constant
WOAS.name (contracts/token/WOAS.sol#19) should be constant
WOAS.symbol (contracts/token/WOAS.sol#20) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

withdraw(uint256) should be declared external:
    - WOAS.withdraw(uint256) (contracts/token/WOAS.sol#38-43)
totalSupply() should be declared external:
    - WOAS.totalSupply() (contracts/token/WOAS.sol#45-47)
approve(address,uint256) should be declared external:
    - WOAS.approve(address,uint256) (contracts/token/WOAS.sol#49-53)
transfer(address,uint256) should be declared external:
    - WOAS.transfer(address,uint256) (contracts/token/WOAS.sol#55-57)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external

renounceOwnership() should be declared external:
    - NFTBridgeMainchain.renounceOwnership() (contracts/nft-bridge/NFTBridgeMainchain.sol#164-166)
    - NFTBridgeSidechain.renounceOwnership() (contracts/nft-bridge/NFTBridgeSidechain.sol#249-251)
    - Ownable.renounceOwnership() (node_modules/@openzeppelin/contracts/access/Ownable.sol#54-56)
name() should be declared external:
    - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#62-64)
symbol() should be declared external:
    - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#70-72)
decimals() should be declared external:
    - ERC20.decimals() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#87-89)
totalSupply() should be declared external:
    - ERC20.totalSupply() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#94-96)
balanceOf(address) should be declared external:
    - ERC20.balanceOf(address) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#101-103)
transfer(address,uint256) should be declared external:
    - ERC20.transfer(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#113-117)
approve(address,uint256) should be declared external:
    - ERC20.approve(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#136-140)
transferFrom(address,address,uint256) should be declared external:
    - ERC20.transferFrom(address,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#158-167)
increaseAllowance(address,uint256) should be declared external:
    - ERC20.increaseAllowance(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#181-185)
decreaseAllowance(address,uint256) should be declared external:
    - ERC20.decreaseAllowance(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#201-210)
name() should be declared external:
    - ERC721.name() (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#79-81)
symbol() should be declared external:
    - ERC721.symbol() (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#86-88)
tokenURI(uint256) should be declared external:
    - ERC721.tokenURI(uint256) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#93-98)
approve(address,uint256) should be declared external:
    - ERC721.approve(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#112-122)
setApprovalForAll(address,bool) should be declared external:
    - ERC721.setApprovalForAll(address,bool) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#136-138)
transferFrom(address,address,uint256) should be declared external:
    - ERC721.transferFrom(address,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#150-159)
safeTransferFrom(address,address,uint256) should be declared external:
    - ERC721.safeTransferFrom(address,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#164-170)
tokenOfOwnerByIndex(address,uint256) should be declared external:
    - ERC721Enumerable.tokenOfOwnerByIndex(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol#37-40)
tokenByIndex(uint256) should be declared external:
    - ERC721Enumerable.tokenByIndex(uint256) (node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol#52-55)
isFirstBlock() should be declared external:
    - Environment.isFirstBlock() (contracts/Environment.sol#69-71)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
. analyzed (42 contracts with 78 detectors), 210 result(s) found
```

oasys-optimism

```
L1BuildDeposit.build(address) (contracts/oasys/L1/build/L1BuildDeposit.sol#109-117) uses a dangerous strict equality:
    - require(bool,string)(_buildBlock[_builder] == 0,already built by builder) (contracts/oasys/L1/build/L1BuildDeposit.sol#112)
L1BuildDeposit.withdraw(address,uint256) (contracts/oasys/L1/build/L1BuildDeposit.sol#88-103) uses a dangerous strict equality:
    - require(bool,string)(buildBlock == 0 || buildBlock + lockedBlock < block.number,while OAS locked) (contracts/oasys/L1/build/L1BuildDeposit.sol#94)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

OasysStateCommitmentChain._updateVerifiedIndex(uint256).verifiedCount (contracts/oasys/L1/rollup/OasysStateCommitmentChain.sol#191) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

Reentrancy in L1BuildDeposit.withdraw(address,uint256) (contracts/oasys/L1/build/L1BuildDeposit.sol#88-103):
    External calls:
    - (success) = depositer.call{value: _amount}() (contracts/oasys/L1/build/L1BuildDeposit.sol#99)
    Event emitted after the call(s):
    - Withdrawal(_builder,depositer,_amount) (contracts/oasys/L1/build/L1BuildDeposit.sol#102)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

OasysStateCommitmentChain._updateVerifiedIndex(uint256) (contracts/oasys/L1/rollup/OasysStateCommitmentChain.sol#187-206) has costly operations inside a loop:
    - nextIndex ++ (contracts/oasys/L1/rollup/OasysStateCommitmentChain.sol#201)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

Reentrancy in WOAS.withdraw(uint256) (contracts/oasys/L1/token/WOAS.sol#38-43):
    External calls:
    - msg.sender.transfer(wad) (contracts/oasys/L1/token/WOAS.sol#41)
    Event emitted after the call(s):
    - Withdrawal(msg.sender,wad) (contracts/oasys/L1/token/WOAS.sol#42)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

Variable L1BuildAgent.setStep1Addresses(uint256,address,address,address,address,address)._ctcBatches (contracts/oasys/L1/build/L1BuildAgent.sol#83) is too similar to
L1BuildAgent.setStep2Addresses(uint256,address,address,address)._sccBatches (contracts/oasys/L1/build/L1BuildAgent.sol#104)
Variable L1BuildAgent.constructor(address,address,address,address,address,address,address)._step1Address (contracts/oasys/L1/build/L1BuildAgent.sol#47) is too similar to
L1BuildAgent.constructor(address,address,address,address,address,address,address)._step2Address (contracts/oasys/L1/build/L1BuildAgent.sol#48)
Variable L1BuildAgent.constructor(address,address,address,address,address,address,address)._step1Address (contracts/oasys/L1/build/L1BuildAgent.sol#47) is too similar to
L1BuildAgent.constructor(address,address,address,address,address,address,address)._step3Address (contracts/oasys/L1/build/L1BuildAgent.sol#49)
Variable L1BuildAgent.constructor(address,address,address,address,address,address,address)._step2Address (contracts/oasys/L1/build/L1BuildAgent.sol#48) is too similar to
L1BuildAgent.constructor(address,address,address,address,address,address,address)._step3Address (contracts/oasys/L1/build/L1BuildAgent.sol#49)
Variable L1BuildAgent.constructor(address,address,address,address,address,address,address)._step1Address (contracts/oasys/L1/build/L1BuildAgent.sol#47) is too similar to
L1BuildAgent.constructor(address,address,address,address,address,address,address)._step4Address (contracts/oasys/L1/build/L1BuildAgent.sol#50)
Variable L1BuildAgent.constructor(address,address,address,address,address,address,address)._step2Address (contracts/oasys/L1/build/L1BuildAgent.sol#48) is too similar to
L1BuildAgent.constructor(address,address,address,address,address,address,address)._step4Address (contracts/oasys/L1/build/L1BuildAgent.sol#50)
Variable L1BuildAgent.constructor(address,address,address,address,address,address,address)._step3Address (contracts/oasys/L1/build/L1BuildAgent.sol#49) is too similar to
L1BuildAgent.constructor(address,address,address,address,address,address,address)._step4Address (contracts/oasys/L1/build/L1BuildAgent.sol#50)
Variable L1BuildAgent.step1Address (contracts/oasys/L1/build/L1BuildAgent.sol#21) is too similar to L1BuildAgent.step2Address (contracts/oasys/L1/build/L1BuildAgent.sol#22)
Variable L1BuildAgent.step1Address (contracts/oasys/L1/build/L1BuildAgent.sol#21) is too similar to L1BuildAgent.step3Address (contracts/oasys/L1/build/L1BuildAgent.sol#23)
Variable L1BuildAgent.step2Address (contracts/oasys/L1/build/L1BuildAgent.sol#22) is too similar to L1BuildAgent.step3Address (contracts/oasys/L1/build/L1BuildAgent.sol#23)
Variable L1BuildAgent.step1Address (contracts/oasys/L1/build/L1BuildAgent.sol#21) is too similar to L1BuildAgent.step4Address (contracts/oasys/L1/build/L1BuildAgent.sol#24)
Variable L1BuildAgent.step2Address (contracts/oasys/L1/build/L1BuildAgent.sol#22) is too similar to L1BuildAgent.step4Address (contracts/oasys/L1/build/L1BuildAgent.sol#24)
Variable L1BuildAgent.step3Address (contracts/oasys/L1/build/L1BuildAgent.sol#23) is too similar to L1BuildAgent.step4Address (contracts/oasys/L1/build/L1BuildAgent.sol#24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

WOAS.decimals (contracts/oasys/L1/token/WOAS.sol#21) should be constant
WOAS.name (contracts/oasys/L1/token/WOAS.sol#19) should be constant
WOAS.symbol (contracts/oasys/L1/token/WOAS.sol#20) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

withdraw(uint256) should be declared external:
    - WOAS.withdraw(uint256) (contracts/oasys/L1/token/WOAS.sol#38-43)
totalSupply() should be declared external:
    - WOAS.totalSupply() (contracts/oasys/L1/token/WOAS.sol#45-47)
approve(address,uint256) should be declared external:
    - WOAS.approve(address,uint256) (contracts/oasys/L1/token/WOAS.sol#49-53)
transfer(address,uint256) should be declared external:
    - WOAS.transfer(address,uint256) (contracts/oasys/L1/token/WOAS.sol#55-57)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external

mint(address,uint32) should be declared external:
    - L1StandardERC721.mint(address,uint32) (contracts/oasys/L1/token/L1StandardERC721.sol#73-76)
pause() should be declared external:
    - L1StandardERC721.pause() (contracts/oasys/L1/token/L1StandardERC721.sol#87-90)
unpause() should be declared external:
    - L1StandardERC721.unpause() (contracts/oasys/L1/token/L1StandardERC721.sol#101-104)
mint(address,uint256) should be declared external:
    - L2StandardERC721.mint(address,uint256) (contracts/oasys/L2/token/L2StandardERC721.sol#54-56)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

Code Documentation

1. **Unresolved:** Change all usage of `Unixtime` to `Unix time` to increase readability.
2. **Fixed:** `manages` is misspelled as `maanages` in `L1BuildDeposit` on `L8`. **Update**: fixed on oasys-optimism@31d61c4.
3. **Fixed:** `transfer` is misspelled as `trasfer` in `SOAS`'s revert message on `L147`. **Update:** fixed on oasys-genesis-contract@6024a78.
4. **Fixed:** `Token` misspells `Wrapped` as `Wrapperd` on `L12` and `L21`. **Update:** fixed on oasys-genesis-contract@6024a78.
5. **Fixed:** `IStakeManager`'s `deactivateValidator` comment should be changed from “Change the validator status to disable” to “Change the validator status to disabled.” **Update:** fixed on oasys-genesis-contract@6024a78.
6. **Unresolved:** The Oasys Technical Materials uses the phrasing "Verse-Layer, with Incredibly High UX." It is unclear what is meant by "High UX."
7. **Unresolved:** Some sentences in doc-hub-layer are confusing. To make the above statements more readable, the selected validator should be explicitly named. For instance, the validator currently proposing blocks could be named a "Block Producer". For example:
    - "Block generation is performed by a node called the validator"
    - “If a validator is out of service for some reason, the next validator performs the block generation work that the validator has failed to do.”
8. **Fixed**: Some missing comments were linted in the build process. **Update**: fixed on oasys-optimism@b0b82a0:

```
Comments Error: Function: (getNamedAddresses(uint256)), returnParam: (_1) is missing @return comment
   @ L1BuildAgent
   --> contracts/oasys/L1/build/L1BuildAgent.sol

Comments Error: Function: (getBuildBlock(address)), returnParam: (_0) is missing @return comment
   @ L1BuildDeposit
   --> contracts/oasys/L1/build/L1BuildDeposit.sol

Comments Error: Function: (getDepositAmount(address,address)), returnParam: (_0) is missing @return comment
   @ L1BuildDeposit
   --> contracts/oasys/L1/build/L1BuildDeposit.sol

Comments Error: Function: (getDepositTotal(address)), returnParam: (_0) is missing @return comment
   @ L1BuildDeposit
   --> contracts/oasys/L1/build/L1BuildDeposit.sol
```

Adherence to Best Practices

1. **Acknowledged:** These state variables are expected to be initialized in the constructor and not to be changed in the contract's life. Consider labeling these state variables as `immutable` (available from Solidity v0.6.5). **Update:** The Oasys team states that Oasys validator's will be hard-coded with the contract’s bytecode and that this makes the contract's `constructor`s unexecutable. Therefore, the `immutable` keyword will no longer be available. From Oasys team: *The bytecodes in* `L1BuildDeposit.sol` *and* `NFTBridgeRelayer.sol` *are hardcoded into the Oasys validator and cannot be immutable (*`versebuilder.go#L69`, `nftBridge.go#L38`*).*
    - `L1BuildDeposit.sol`
        - `uint256 public requiredAmount`
        - `uint256 public lockedBlock`
        - `uint256 public allowlistAddress`
        - `uint256 public agentAddress`
    - `NFTBridgeRelayer.sol`
        - `address public mainchainBridge`
        - `address public sidechainBridge`
    - `SidechainERC721.sol`. **Update:** fixed on oasys-genesis-contract@9dc09fe.
        - `uint256 private mainchainId`
        - `address private mainchainERC721`
    - `StakeManager.sol`
        - `IEnvironment public environment`
        - `IAllowlist public allowlist`
    - `SOAS.sol`
        - `address public staking`
2. **Fixed:** Naming convention: it is recommended to name internal and private elements starting with an underscore. **Update:** fixed on oasys-optimism@36a6961, and oasys-genesis-contract@9dc09fe:
    - `L1BuildDeposit.sol`
        - `mapping(address => uint256) private depositTotal`

-
    - `mapping(address => mapping(address => uint256)) private depositAmount`
    - `mapping(address => uint256) private buildBlock`
- `NFTBridgeMainchain.sol`
    - `DepositInfo[] private depositInfos`
- `NFTBridgeSidechain.sol`
    - `mapping(uint256 => mapping(address => address)) private erc721Map`
    - `mapping(uint256 => mapping(uint256 => bool)) private depositIndexes`
    - `mapping(address => mapping(uint256 => uint256)) private depositIndexMap`
    - `WithdrawalInfo[] private withdrawalInfos`
- `SidechainERC721.sol`
    - `uint256 private mainchainId`
    - `address private mainchainERC721`
- `Signers.sol`
    - `address[] private signers`
    - `function recoverSigner(...) private pure`

3. **Fixed:** Change `L1BuildDeposit` `calldata` on `L93` from `new bytes(0)` to `""` to save gas. **Update:** fixed on oasys-optimism@36a6961.
4. **Fixed:** Event indexing makes tracking contract state changes easier for explorers and end-users. Add indexing to the following events. **Update:** fixed on oasys-genesis-contract@9dc09fe.
    - `Signer`'s `SignerAdded`, `SignerRemoved`, and `ThresholdUpdated`.
5. **Fixed:** Using a list to track `signers` is gas inefficient as checking whether a given `address` is part of the `signers` list takes linear time. Change `signers` to be a mapping, which will allow `signers` `address` inclusion check to be performed in constant time. **Update:** fixed on oasys-genesis-contract@9dc09fe.
6. **Acknowledged:** Add a `Status` `enum` to `DepositInfo` to denote a pending state instead of using `DepositInfo.mainTo` to increase code readability. **Update:** from Oasys team: *We do not use enum since it is sufficient to determine whether the address is address(0) or not.*
7. **Fixed:** Some public functions are not called internally and should be declared external. **Update:** fixed on oasys-genesis-contract@6024a78, oasys-genesis-contract@6daaf50, and oasys-genesis-contract@9dc09fe.
    - `SidechainERC721.mint(...)` on `L46`
    - `SidechainERC721.burn(...)` on `L52`
    - `Environment.isFirstBlock(...)` on `L69`
    - `WOAS.transfer(...)` on `L55`
    - `WOAS.approve(...)` on `L49`
    - `WOAS.totalSupply(...)` on `L45`
    - `WOAS.withdraw(...)` on `L38`
8. **Fixed:** Whenever iterating over a storage list, save its length in memory prior to iterating over it to minimize `SLOAD` operations. For example `StakeManager`'s `updateValidatorBlocks` `operator` iteration should be changed from:

   ```
   for (uint256 i = 0; i < operators.length; i++) {
   ```

   to

   ```
   uint256 operatorsLength = operators.length;
   for (uint256 i = 0; i < operatorsLength; i++) {
   ```

   Update: fixed on oasys-genesis-contract@6024a78.
9. **Fixed:** Add indexing to `IAllowlist`'s events to make it easier for dapps and end-users to search for them. **Update:** fixed on oasys-genesis-contract@6024a78.
10. **Fixed** Consider whether all of `ClaimInfo`s numerical fields need to be `uint256` as using a smaller unsigned integers such as `uint128` would save gas. **Update:** fixed on oasys-genesis-contract@6024a78.
11. **Fixed:** Change all non-function calling `call`s to use `""` instead of `new bytes(0)` to save gas. **Update:** fixed on oasys-genesis-contract@6024a78.
12. **Acknowledged:** Change `SOAS`'s and `WOAS`'s ERC20 parameters to be configurable instead of hardcoded, to accommodate future configuration changes without requiring smart contract code changes. **Update:** The Oasys team states that `WOAS`'s and `SOAS`'s parameters do not need to be configurable as there are no plans addressed as there are no plans to make further changes to or to reuse the contracts. inherit them
13. **Fixed** Modify `WOAS`'s `transferFrom` function to check if `src` and `dst` are the same prior to performing storage modifying operations on `balanceOf` to save gas. **Update:** fixed on oasys-genesis-contract@6daaf50.
14. **Fixed:** The following variables should be declared `constant`s in order to save gas:
    - `WOAS.decimals` on `L21`
    - `WOAS.name` on `L19`
    - `WOAS.symbol` on `L20`

    Update: fixed on oasys-genesis-contract@6daaf50.
15. **Mitigated:** Change all `require` statements to use custom `errors` to save gas. **Update:** The majority of `require` statements were removed in favor of customer `error`s. Commit: oasys-genesis-contract@6024a78.
16. **Fixed:** Remove `Environment`'s `_getNext` functions conditional on `L125` as it is a no-op. **Update:** Fixed on oasys-genesis-contract@6024a78.
17. **Acknowledged:** Change `Token` to not differentiate between `WOAS` and `SOAS`, as there is no need to, as they are both `ERC20` compliant tokens. **Update:** The Oasys team

states that WOAS and SOAS must be distinguished because their roles are completely different.

18. **Fixed:** Change EnvironmentValue's validate function's hardcoded numbers to be constants to increase readability. **Update:** Fixed on oasys-genesis-contract@6024a78.
19. **Acknowledged:** In core/blockchain.go on L133 there is a commented out TODO message. Either complete the TODO task, verify that it has already been completed, or verify that it is no longer needed. Then remove the TODO comment. **Update:** The Oasys team states that the TODO originally exists in go-ethereum, the source of the fork.
20. **Acknowledged:** In consensus/oasys/contract.go's getNextValidators(...) consider adding more sanity checks to the function to handle unexpected situations. For example, the following check could be added:

    - If the current number of validators is larger than a pre-specified N number of validators, then something unexpected has happened and the Oasys blockchain pauses.

    Update: The Oasys team states that at this time, there are no plans to set any restrictions other than having at least 10 million tokens deposited.
21. **Fixed:** Remove or uncomment all commented code in the Oasys Validator client to increase readability. For example most of consensus/oasys/snapshot_test.go is commented out and it is unclear whether those tests are meant to be run.
22. **Unresolved:** Consider removing support for past Ethereum forks in Oasys as the current genesis/mainnet.json has all fork starting blocks set to 0. If fork support is needed, add technical documentation explaining why.

Test Results

Test Suite Results

The test suites for the projects do not implement tests for the scoped functionalities.

Update: test suite provided for the re-audit phase. Please check failing tests for oasys-optimism.

```
=== RUN   TestInitializeSystemContracts
--- PASS: TestInitializeSystemContracts (0.01s)
=== RUN   TestSlash
--- PASS: TestSlash (0.00s)
=== RUN   TestGetNextValidators
--- PASS: TestGetNextValidators (0.00s)
=== RUN   TestGetRewards
--- PASS: TestGetRewards (0.00s)
=== RUN   TestGetNextEnvironmentValue
--- PASS: TestGetNextEnvironmentValue (0.00s)
=== RUN   TestAddBalanceToStakeManager
--- PASS: TestAddBalanceToStakeManager (0.00s)
=== RUN   TestBackOffTime
--- PASS: TestBackOffTime (0.01s)
=== RUN   TestGetValidatorSchedule
--- PASS: TestGetValidatorSchedule (0.00s)
PASS
ok      github.com/ethereum/go-ethereum/consensus/oasys 0.478s
```

oasys-genesis-contract

```
  Environment
     ✓ initialize() (93ms)
     ✓ value() (3141ms)
     ✓ epochAndValues() (340ms)
    updateValue()
       ✓ startEpoch is past (46ms)
    epoch()
       ✓ simple case (1965ms)
       ✓ update in last block of epoch (309ms)
       ✓ update in first block of epoch (820ms)
       ✓ overwriting the same epoch (2118ms)
       ✓ overwriting the same epoch (1682ms)

  NFTBridgeMainchain
     ✓ depositInfos()
     ✓ transferOwnership()
     ✓ renounceOwnership()
    deposit()
       ✓ normally
       ✓ sideTo is zero address
    rejectDeposit()
       ✓ normally (42ms)
       ✓ invalid chain id (115ms)
       ✓ already rejected (43ms)
    finalizeWithdrawal()
       ✓ normally (42ms)
       ✓ invalid chain id
       ✓ mainTo is zero address
       ✓ already withdraw (39ms)
       ✓ failed token transfer (49ms)
    transferMainchainRelayer()
       ✓ normally
       ✓ invalid nonce
       ✓ invalid to
       ✓ invalid chain id

  NFTBridgeSidechain
     ✓ getSidechainERC721()
     ✓ getWithdrawalInfo() (41ms)
     ✓ transferOwnership()
     ✓ renounceOwnership()
    createSidechainERC721()
       ✓ normally
       ✓ invalid chain id
       ✓ same chain id
       ✓ already exists
    finalizeDeposit()
       ✓ normally
       ✓ invalid chain id
       ✓ sidechain erc721 not found
       ✓ already deposited
       ✓ already minted
    withdraw()
       ✓ normally (42ms)
       ✓ invalid sidechainERC721
    rejectWithdrawal()
       ✓  normally (57ms)
       ✓ invalid chain id (48ms)
       ✓ already rejected (56ms)
    transferSidechainRelayer()
       ✓ normally
       ✓ invalid nonce
       ✓ invalid to
       ✓ invalid chain id

  Signers
    constructor()
       ✓ normally
       ✓ signer is zero address
       ✓ duplicate signer
       ✓ threshold is zero
    verifySignatures()
       ✓ signature expired
```

```
      ✓ invalid signatures length
      ✓ invalid chain id
      ✓ invalid expiration
      ✓ below Threshold
    addSigner()
      ✓ normally
      ✓ signer is zero address
      ✓ invalid nonce
      ✓ invalid to
      ✓ already added
    removeSigner()
      ✓ normally
      ✓ invalid nonce
      ✓ invalid to
      ✓ signer shortage
    updateThreshold()
      ✓ normally
      ✓ threshold is zero
      ✓ invalid nonce
      ✓ invalid to
      ✓ signer shortage

  StakeManager
    ✓ initialize() (85ms)
    validator owner or operator functions
      ✓ joinValidator()
      ✓ updateOperator() (41ms)
      ✓ deactivateValidator() and activateValidator() (65ms)
      ✓ updateCommissionRate() (176ms)
      ✓ claimCommissions() (643ms)
    staker functions
      ✓ stake() (1140ms)
      ✓ unstake() and claimUnstakes() (3042ms)
    rewards and commissions
      ✓ when operating ratio is 100% (7199ms)
      ✓ when operating ratio is 50% (1729ms)
      ✓ when operating ratio is 0% (2435ms)
    current validator functions
      ✓ slash() (1668ms)
      ✓ updateValidators() (606ms)
    view functions
      ✓ getCurrentValidators() (191ms)
      ✓ getValidators()
      ✓ getStakers() (194ms)
      ✓ getValidatorInfo() (603ms)
      ✓ getStakerInfo() (352ms)
      ✓ getValidatorStakes() (983ms)
      ✓ getStakerStakes() (1739ms)
      ✓ getBlockAndSlashes() (1103ms)
      ✓ getTotalRewards() (1144ms)


  93 passing (52s)
```

oasys-optimism

```
  L1BuildDeposit
    ✓ getDepositTotal()
    ✓ getDepositAmount()
    ✓ getBuildBlock()
    deposit()
      ✓ normally
      ✓ builder is zero address
      ✓ no OAS
      ✓ builder not allowed
      ✓ over deposit amount
    withdraw()
      1) normally
      ✓ builder is zero address
      ✓ amount is zero
      2) while OAS locked
      ✓ immediate withdraw if not built
      3) deposit amount shortage
    build()
      ✓ normally
      ✓ call from non-agent
      ✓ deposit amount shortage
      ✓ already built


  765 passing (1m)
  5 pending
  3 failing

  1) L1BuildDeposit
       withdraw()
         normally:
     MethodNotFoundError: Method hardhat_mine not found
      at HardhatModule.processRequest (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-github/node_modules/hardhat/src/internal/hardhat-
network/provider/modules/hardhat.ts:110:11)
      at HardhatNetworkProvider._send (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-github/node_modules/hardhat/src/internal/hardhat-
network/provider/provider.ts:199:35)
      at runMicrotasks (<anonymous>)
      at processTicksAndRejections (node:internal/process/task_queues:96:5)
      at runNextTicks (node:internal/process/task_queues:65:3)
      at listOnTimeout (node:internal/timers:528:9)
      at processTimers (node:internal/timers:502:7)
      at async HardhatNetworkProvider.request (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-
github/node_modules/hardhat/src/internal/hardhat-network/provider/provider.ts:106:18)
      at async Context.<anonymous> (test/contracts/oasys/L1/build/L1BuildDeposit.spec.ts:88:7)

  2) L1BuildDeposit
       withdraw()
         while OAS locked:
     MethodNotFoundError: Method hardhat_mine not found
      at HardhatModule.processRequest (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-github/node_modules/hardhat/src/internal/hardhat-
network/provider/modules/hardhat.ts:110:11)
      at HardhatNetworkProvider._send (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-github/node_modules/hardhat/src/internal/hardhat-
network/provider/provider.ts:199:35)
      at runMicrotasks (<anonymous>)
      at processTicksAndRejections (node:internal/process/task_queues:96:5)
      at runNextTicks (node:internal/process/task_queues:65:3)
      at listOnTimeout (node:internal/timers:528:9)
      at processTimers (node:internal/timers:502:7)
      at async HardhatNetworkProvider.request (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-
github/node_modules/hardhat/src/internal/hardhat-network/provider/provider.ts:106:18)
      at async Context.<anonymous> (test/contracts/oasys/L1/build/L1BuildDeposit.spec.ts:111:7)

  3) L1BuildDeposit
       withdraw()
         deposit amount shortage:
     MethodNotFoundError: Method hardhat_mine not found
      at HardhatModule.processRequest (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-github/node_modules/hardhat/src/internal/hardhat-
network/provider/modules/hardhat.ts:110:11)
      at HardhatNetworkProvider._send (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-github/node_modules/hardhat/src/internal/hardhat-
network/provider/provider.ts:199:35)
      at runMicrotasks (<anonymous>)
      at processTicksAndRejections (node:internal/process/task_queues:96:5)
      at runNextTicks (node:internal/process/task_queues:65:3)
      at listOnTimeout (node:internal/timers:528:9)
      at processTimers (node:internal/timers:502:7)
      at async HardhatNetworkProvider.request (/Users/guillermoescobero/Documents/audits/at-1178-oasys2/reaudit/oasysgames-oasys-optimism-31d61c426ccb161093010680a9cbec654479dda6-
github/node_modules/hardhat/src/internal/hardhat-network/provider/provider.ts:106:18)
      at async Context.<anonymous> (test/contracts/oasys/L1/build/L1BuildDeposit.spec.ts:124:7)
```

Code Coverage

Only files related to Oasys were included in this coverage analysis. It shows a zero percentage of coverage in both repositories as no tests are implemented for these system features.

Update: Coverage of scoped files increased as the test suite was implemented during the fixes phase. However, Quantstamp recommends coverage of at least 90% in the project before deploying.

oasys-genesis-contract

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	98	78.85	97.73	98.14	
Environment.sol	91.18	91.67	90.91	91.18	92,93,114
IEnvironment.sol	100	100	100	100	
IStakeManager.sol	100	100	100	100	
StakeManager.sol	100	75	100	100	
System.sol	100	75	100	100	
contracts/lib/	92.83	81.82	95.92	93.09	
Allowlist.sol	45.45	33.33	60	45.45	… 51,53,60,69
Constants.sol	100	100	100	100	
EnvironmentValue.sol	100	50	100	100	
IAllowlist.sol	100	100	100	100	
Math.sol	100	100	100	100	
Staker.sol	97.5	96.88	100	97.44	239,240
Token.sol	92.31	66.67	100	91.67	71
UpdateHistories.sol	100	100	100	100	
Validator.sol	96.92	91.18	100	100	
contracts/nft-bridge/	98.64	87.5	97.37	98.69	
INFTBridgeMainchain.sol	100	100	100	100	
INFTBridgeSidechain.sol	100	100	100	100	
NFTBridgeMainchain.sol	100	100	100	100	
NFTBridgeRelayer.sol	100	64.29	100	100	
NFTBridgeSidechain.sol	100	100	100	100	
SidechainERC721.sol	88.89	100	83.33	88.89	76
Signers.sol	98.25	85	100	98.36	188
contracts/test/	80	100	60	80	
TestERC20.sol	0	100	0	0	12
TestERC721.sol	100	100	100	100	
contracts/token/	0	0	0	0	
SOAS.sol	0	0	0	0	… 130,132,147
WOAS.sol	0	0	0	0	… 70,71,73,75
All files	**87.46**	**74.83**	**86.67**	**87.73**	

oasys-optimism

Tests failing.

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

```
0023fa202992338e6ccb0930726fb545b539605fde6f7777e984a8c2b5582c54 ./oasys-genesis-contract/contracts/IStakeManager.sol

64e5eed3e99ac05eb84b9ad5b26bac8575fab3f55774d2c7c12e6d1b5f5acafb ./oasys-genesis-contract/contracts/System.sol

976832655e78982587d537eea961899b3575f394e2cdd157da150887f28abc27 ./oasys-genesis-contract/contracts/StakeManager.sol

0ae1fccd35ff351a6ba1ac675d6c5119eeb8c7c2f1ca1d03a1666cef91c76d35 ./oasys-genesis-contract/contracts/IEnvironment.sol

4b70175930124bc543db0ca86d19bc41a50e275d138b7ca637259eb7d4c31688 ./oasys-genesis-contract/contracts/Environment.sol

2b499b683ec34fd7d06f85fd0fcb65937f0beb909a13f5ae75b756cb98b67759 ./oasys-genesis-contract/contracts/lib/EnvironmentValue.sol

1fde6f9d5954d38ba9891b1923b37ca2ba21e6d43226838cc5306c8b96f26deb ./oasys-genesis-contract/contracts/lib/Token.sol

949c5c73a6d0c7f3918747f559251d6d33b31c6b7c523715fa6f3b0cfa42d220 ./oasys-genesis-contract/contracts/lib/IAllowlist.sol

733a0f442e8d5b8890eed230d28ea8d5d90c16727f97a538f789f5c7afbceeab ./oasys-genesis-contract/contracts/lib/Staker.sol

c787097f21ecd517b37fa0e3f2bd9d6fa667883f58a2715e2c64240ed6f9d0ac ./oasys-genesis-contract/contracts/lib/Validator.sol

b68d3284d2f3d062dd1fcf0052c2f0187c5f0415c9ffa94e88517f36bedfaf6a ./oasys-genesis-contract/contracts/lib/Allowlist.sol

e8e681154e17a86dea8a83ad0d78b27fc9379d6c63981c2e0f56c6900c0d5f26 ./oasys-genesis-contract/contracts/lib/Constants.sol

d49527857fc6adcbf785f69f4f8c59b9012eb7837575467ab35147541d5e32eb ./oasys-genesis-contract/contracts/lib/Math.sol

c9aff8c8b969c7d027a7cd79eba35d097c33498ae56880f51aa37d4768bad061 ./oasys-genesis-contract/contracts/lib/UpdateHistories.sol

31ac24ffa9b4d5075fa84f7e932c7245eec644812185625e69dd5b760b0a33d7 ./oasys-genesis-contract/contracts/nft-bridge/NFTBridgeMainchain.sol

3766c0d0b6a09986386b7a45eaac30d69045e1ac6fcfe626c5069fd33c2194b4 ./oasys-genesis-contract/contracts/nft-bridge/SidechainERC721.sol

b79afe46a80657b15a6d7dfd1ab9d753d6ef9c61d66f50899770d52722bea636 ./oasys-genesis-contract/contracts/nft-bridge/INFTBridgeSidechain.sol

d266fa0c7b402cf38afe71c9d38df1097aea17b532be884026025a75ac8c2b17 ./oasys-genesis-contract/contracts/Signers.sol

00ee6424bf3b09e013680cc760bfe186961daffc1694b6244ae06d23ce2b33e3 ./oasys-genesis-contract/contracts/nft-bridge/NFTBridgeSidechain.sol

81a2c7172db32098b7986450d2e2b30fd81d74cbeade21ef9f6a5d33577150b6 ./oasys-genesis-contract/contracts/nft-bridge/INFTBridgeMainchain.sol

6904960cc9e20b114cb9ad37d5b62db90fc8acbb0cc2483235415682dceb6dcd ./oasys-genesis-contract/contracts/nft-bridge/NFTBridgeRelayer.sol

5b24958a1715221af6dc0d91a7fe5ea4b823bc6afe1b5cf497801ecc14ed0712 ./oasys-genesis-contract/contracts/token/WOAS.sol

4f12c487792b647c167842104436e7ea617f9cdad3f96eb16dc6c892a770a8f9 ./oasys-genesis-contract/contracts/token/SOAS.sol

16b51bc8c6a0367c0c08d74783829be297a051461a74dfe2c36a09fe1abdb24a ./oasys-validator/core/state_processor.go

11470357f1234a4544307e29d537c1e699df39a789f10680b344107f4990ca44 ./oasys-validator/core/receipt_processor.go

acb7400c44b30e91bf9c0285edc73f3ed1d2f287a05c7b6b3a164b386d20740b ./oasys-validator/core/blockchain.go

1cf557919df2b74786e1731054f8f37cb6f0a28860bb668d893ec5c52e4aedb0 ./oasys-validator/consensus/oasys/api.go

9032464769489ef38f260df87aaf9f24054b444aa8bd39091751ea8c98fd973e ./oasys-validator/consensus/oasys/contract.go

1a285723639f45bc030516830588496ed87f2a09f53c8cf123403428d5f87d3e ./oasys-validator/consensus/oasys/oasys.go

0a95c1a3fc0148ab7afbccb1c686618bb6a578d9878b5e74ccee053aa64c490e ./oasys-validator/consensus/oasys/snapshot.go

bb854018a829d978f72ac32d3d473c81c798aa9b0fa1cb5b5d10cc4ed1bea192 ./oasys-
optimism/packages/contracts/contracts/oasys/L1/build/L1BuildDeposit.sol
```

Tests

```
13351506936bb3e405499f5c851e5523802606f6de6f98a98741203bac246751 ./oasys-genesis-contract/test/helpers.ts

7fdfcebf771f9f54546fd550ab291e68055c14fb8de3d3c5ec05913373532046 ./oasys-genesis-contract/test/StakeManager.spec.ts

a0e738dc0b79e0342a36fa0331119ae093caa815565c4887c4ecdc02e40f14d0 ./oasys-genesis-contract/test/Environment.spec.ts

6560f9fb0a2f943105cec958aef3c0bfdaada5bf5719f7e6c5c6fe1817f6d52e ./oasys-genesis-contract/test/lib/Allowlist.spec.ts

79c76f051c453fe34f42df9cb149c0a1c9611b5def7622e5a84b80da82ca97d0 ./oasys-genesis-contract/test/nft-bridge/NFTBridgeMainchain.spec.ts

ef4636109153cd00da5ba099c47f2ab5a78767b5584e7fecf8f1a889a8bd2b52 ./oasys-genesis-contract/test/nft-bridge/Signers.spec.ts

c8ee45dcd16bb7d665a20869087b20fe383026d06b27a698b16ae86c2130df6e ./oasys-genesis-contract/test/nft-bridge/NFTBridgeSidechain.spec.ts

75f5e70f2cabdbc6d1b103c6b120f1f1df36184d2372eade9ca49b6c120a8a4e ./oasys-genesis-contract/test/token/WOAS.spec.ts

47e58cda08e3678373baff10cc22dd3ece62cb5f52fc3f3b8448428478f31ac1 ./oasys-genesis-contract/test/token/SOAS.spec.ts

a255a79aad352116482e545fb6431cbe60b5dac1abf12d4aa2319715e4755032 ./oasys-validator/consensus/oasys/oasys_test.go

430cee5e0266af99d05102f98fb73a0c153406ed02a10e56bb39f5b268ea385f ./oasys-validator/consensus/oasys/contract_test.go

6e64a5e66a7bf5012e146cae4b45dab5aacfc4ac0a2dee7035e650a53bcf173f ./oasys-
optimism/packages/contracts/test/contracts/oasys/L1/build/L1BuildDeposit.spec.ts
```

Changelog

- 2022-06-20 - Initial report
- 2022-07-01 - Final report

About Quantstamp

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

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

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

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