Quantstamp Security Assessment Certificate

October 18th 2022 — Quantstamp Verified

Moloch V3

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

Executive Summary

Type	DAO
Auditors	Ibrahim Abouzied, Auditing Engineer Danny Aksenov, Security Auditor Fatemeh Heidari, Security Auditor
Timeline	2022-09-16 through 2022-09-23
EVM	Arrow Glacier
Languages	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	Baal Docs
Documentation Quality	High
Test Quality	High

Source Code

Repository	Commit
HausDAO/Baal	5b64eab None
HausDAO/Baal	84b7673 None

Total Issues	**16** (7 Resolved)
High Risk Issues	**1** (1 Resolved)
Medium Risk Issues	**1** (1 Resolved)
Low Risk Issues	**5** (3 Resolved)
Informational Risk Issues	**7** (2 Resolved)
Undetermined Risk Issues	**2** (0 Resolved)

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
Low Risk	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low-impact in view of the client's business circumstances.
Informational	The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
? Undetermined	The impact of the issue is uncertain.
Unresolved	Acknowledged the existence of the risk, and decided to accept it without engaging in special efforts to control it.
Acknowledged	The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).
Fixed	Adjusted program implementation, requirements or constraints to eliminate the risk.
Mitigated	Implemented actions to minimize the impact or likelihood of the risk.

Summary of Findings

Moloch V3 brings Baal, a DAO template that allows DAOs to easily deploy on-chain governance and integrate with a new or existing treasury. One of the more notable features in V3 is the use of Shamans: external contracts that the DAO approves to carry out DAO functions without a DAO proposal. This includes actions such as minting/burning shares and loot, adjusting governance parameters, and toggling share/loot transferability.
It is important to note that due to Baal's composability, it is expected that DAOs will choose to integrate Baal with a variety of Shaman smart contracts. With the infinite possibilities of Shaman implementations, the scope of this audit can only assess the security of the base Baal template. DAOs should make their own assessments of a smart contract's security before making it a Shaman.
A few vulnerabilities were discovered, though we found the protocol to be well-designed as a whole, with most of the vulnerabilities requiring minimal changes to address. We recommend that the HausDAO team address these vulnerabilities before making Moloch V3 available to users.
A cursory view of the unit tests indicates a strong testing suite, but this cannot be confirmed as we were unable to run coverage analysis on the unit tests. We encourage the HausDAO team to instrument unit test coverage to validate strong code coverage.

ID	Description	Severity	Status
QSP-1	Checkpoints May Not Be Written Correctly	⇈ High	Fixed
QSP-2	Baal Inherits From Non-Upgradeable Contracts	^ Medium	Fixed
QSP-3	Integer Overflow / Underflow	⌄ Low	Fixed
QSP-4	Missing Input Validation	⌄ Low	Mitigated
QSP-5	Ownership Can Be Renounced	⌄ Low	Acknowledged
QSP-6	Shamans Can Be an EOA Address	⌄ Low	Acknowledged
QSP-7	Signed Votes Do Not Expire	⌄ Low	Fixed
QSP-8	Application Monitoring Can Be Improved by Emitting More Events	○ Informational	Fixed
QSP-9	`setAdminConfig` Always Emits Two Events Even if State Is Not Changed.	○ Informational	Fixed
QSP-10	Risk of Killing Upgrades	○ Informational	Acknowledged
QSP-11	Clone-and-Own	○ Informational	Acknowledged
QSP-12	A DAO's Safety Is Dependent on the Safety of Its Shamans	○ Informational	Acknowledged
QSP-13	Upgradability	○ Informational	Acknowledged
QSP-14	`msg.sender` Can Be Overridden.	○ Informational	Acknowledged
QSP-15	External Calls to Malicious Contracts	? Undetermined	Acknowledged
QSP-16	Proposals Can Pass without a Valid Sponsor	? Undetermined	Acknowledged

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:
The audit was performed on the following files only: `contracts/*`
The audit excluded the following files: `contracts/mock/*`

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 .`

Findings

QSP-1 Checkpoints May Not Be Written Correctly

Severity: *High Risk*

Status: Fixed

File(s) affected: `BaalVotes.sol`

Description: `BaalVotes` has a `_writeCheckpoint()` function that is used to track changes in the number of votes and delegates. It incorrectly assumes that `getCheckpoint()` will return the data type `Checkpoint storage` when it actually returns `Checkpoint memory`. The two functions have been reproduced below.

```
function _writeCheckpoint(
    address delegatee,
    uint256 nCheckpoints,
    uint256 oldVotes,
    uint256 newVotes
) private {
    uint32 timeStamp = uint32(block.timestamp);

    unchecked {
```

```
        if (
            nCheckpoints != 0 &&
            getCheckpoint(delegatee, nCheckpoints - 1).fromTimeStamp ==
            timeStamp
        ) {
            getCheckpoint(delegatee, nCheckpoints - 1).votes = newVotes; // <- This change will not persist.
        } else {
            checkpoints[delegatee][nCheckpoints] = Checkpoint(
                timeStamp,
                newVotes
            );
            numCheckpoints[delegatee] = nCheckpoints + 1;
        }
    }

    emit DelegateVotesChanged(delegatee, oldVotes, newVotes);
}

function getCheckpoint(address delegatee, uint256 nCheckpoints)
    public
    view
    virtual
    returns (Checkpoint memory)
{
    return checkpoints[delegatee][nCheckpoints];
}
```

Recommendation: Update the code segment so that `_writeCheckpoint()` updates the storage address.

Update: `getCheckoutpoint()` has been replaced by `checkpoints[delegatee][nCheckpoints - 1]` in `_writeCheckpoint()` function.

QSP-2 Baal Inherits From Non-Upgradeable Contracts

Severity: *Medium Risk*

Status: Fixed

File(s) affected: `Baal.sol`

Description: `Baal` is intended to be an upgradeable contract, as indicated by its use of a `setUp()` function. However, it inherits from non-upgradeable contracts `EIP712` and `ReentrancyGuard`.
Additionally, `versionRecipient` is initialized outside of the `setUp()` function and will not be initialized in any proxies.

Recommendation: Replace the non-upgradeable contracts with their upgradeable counterparts from Open-Zeppelin.
Assign `versionRecipient` in the `setUp()` function.

Update: Contracts in question have been replaced by their upgradeable counter-parts and are being initialized in the initializer function. `draft-EIP712Upgradeable.sol` can be replaced with `EIP712Upgradeable.sol` as OZ has recently finalized their EIP712 implementation.

QSP-3 Integer Overflow / Underflow

Severity: *Low Risk*

Status: Fixed

File(s) affected: `Baal.sol`

Related Issue(s): SWC-101

Description: Unchecked operations can lead to overflow/underflow. `proposalCount` in the `Baal.submitProposal(..)` function is increased by one for every proposal submitted. If `proposalOffering` is small or zero, it is possible for `submitProposal(..)` to be called repeatedly until the `proposalCount` reaches its max value and overflows.

Recommendation: Remove unchecked operations on `proposalCount` to prevent overflow.

Update: The unchecked operation has been removed.

QSP-4 Missing Input Validation

Severity: *Low Risk*

Status: Mitigated

File(s) affected: `Baal.sol`, `BaalSummoner.sol`, `BaalVotes.sol`, `LootERC20`, `SharesERC20`

Related Issue(s): SWC-123

Description: It is important to validate inputs, even if they only come from trusted addresses, to avoid human error. Some functions do not validate their inputs, which can result in unexpected behavior by the contracts. A non-exhaustive list includes:

- `Baal.setUp()`: Validate that all addresses are non-zero.
- `Baal.submitProposal()`: Validate that `expiration` is not less than `block.timestamp`.
- `Baal.setGovernanceConfig()`
  . Validate that `quorum` is a value between 0..100.
  . Validate that `minRetention` is a value between 0..100.
  . Validate that `sponsor` is a value below `sharesToken.totalSupply()`.
- `BaalSummoner.constructor()`: Validate that all addresses are non-zero.
- `BaalVotes.delegateBySig()`: Validate that the signer is a non-zero address.
- `LootERC20.setUp()`: Validate that `name_` and `_symbol` are non-empty strings.
- `SharesERC20.setUp()`: Validate that `name_` and `_symbol` are non-empty strings.

Recommendation: We recommend adding the relevant checks.

Update: All of the missing input validation checks have been implemented except for one in `Baal.setGovernanceConfig()`:
`require(quorum >= 0 && minRetention <= 100, 'bad quorum');`
should be replaced with:
`require(quorum >= 0 && quorum <= 100, 'bad quorum');`

QSP-5 Ownership Can Be Renounced

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `LootERC20.sol`, `SharesERC20.sol`, `Baal.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. Disable `renounceOwnership()` so that the contract always has an owner.

Update: The team has addressed this in their documentation.

QSP-6 Shamans Can Be an EOA Address

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `Baal.sol`

Description: A Shaman is a separate contract that the DAO approves to make critical changes to the DAO outside of the proposal process. Any address can be approved by the DAO to have Shaman permissions, even if it is an EOA address.

Recommendation: In `setShamans()`, require that the addresses belong to contracts. This can be done by checking the `extcodesize()`.

Update: The team has addressed this in their documentation.

QSP-7 Signed Votes Do Not Expire

Severity: *Low Risk*

Status: Fixed

File(s) affected: `Baal.sol`

Description: Members can submit a vote with an EIP-712 signature. With the current implementation, signed votes are considered to be valid indefinitely and can be submitted at any time within the voting period. With a sufficiently long voting period, it is possible that a user may have changed their mind with regard to their vote.

Recommendation: Allow users to sign their vote with an expiration date after which the vote cannot be submitted.

Update: Signatures are now submitted with an expiry.

QSP-8 Application Monitoring Can Be Improved by Emitting More Events

Severity: *Informational*

Status: Fixed

File(s) affected: `Baal.sol`, `TributeMinion.sol`

Description: In order to validate the proper deployment and initialization of the contracts, it is a good practice to emit events. Also, any important state transitions can be logged, which is beneficial for monitoring the contract, and also tracking eventual bugs or hacks. Below we present a non-exhaustive list of events that could be emitted to improve application management:

- `Baal.lockAdmin(..)`
- `Baal.lockManager(..)`
- `Baal.lockGovernor(..)`
- `TributeMinion.releaseEscrow(..)`

Recommendation: Consider emitting the events.

Update: The team has added the aforementioned events.

QSP-9 `setAdminConfig` Always Emits Two Events Even if State Is Not Changed.

Severity: *Informational*

Status: Fixed

File(s) affected: `Baal.sol`

Description: The `setAdminConfig` function emits `SharesPaused` or `LootPaused` event regardless of whether a state change has actually occurred.

Recommendation: Revise the function to only emit events if the state is changed.

Update: The team has implemented the recommendation.

QSP-10 Risk of Killing Upgrades

Severity: *Informational*

Status: Acknowledged

File(s) affected: `LootERC20.sol`, `SharesERC20.sol`

Description: Both the Loot and Shares tokens make use of the UUPS pattern for upgradeable contracts. One of the drawbacks of using such a pattern is that if a future implementation does not implement the `upgradeTo` function, then upgrades for the tokens have effectively been killed.

Recommendation: Understand the drawbacks of using a UUPS pattern and document the potential risks for users.

Update: The team has addressed this in their documentation.

QSP-11 Clone-and-Own

Severity: *Informational*

Status: Acknowledged

File(s) affected: `BaalVotes.sol`

Description: The clone-and-own approach involves copying and adjusting open source code at one's own discretion. From the development perspective, it is initially beneficial as it reduces the amount of effort. However, from the security perspective, it involves some risks as the code may not follow the best practices, may contain a security vulnerability, or may include intentionally or unintentionally modified upstream libraries. The open source code in question is Compound's governance token.

Recommendation: Rather than the clone-and-own approach, a good industry practice is to use a package manager (e.g., npm) for handling library dependencies. This eliminates the clone-and-own risks yet allows for following best practices, such as using libraries. If the file is cloned anyway, a comment including the repository, the commit hash of the version cloned, and the summary of modifications (if any) should be added. This helps to improve the traceability of the file.

Update: The team has addressed this in their documentation.

QSP-12 A DAO's Safety Is Dependent on the Safety of Its Shamans

Severity: *Informational*

Status: Acknowledged

Description: A Shaman is a separate contract that the DAO approves to make critical changes to the DAO outside of the proposal process. Given that they may have permission to change the contract's configuration, requirements for passing proposals, and mint/burn any user's loot/share tokens, it is important that DAO members thoroughly understand a contract before granting it shaman permissions. Since DAOs are intended to come with their own Shaman contracts, the safety of any particular Shaman contract cannot be assessed and is outside the scope of this audit.

Recommendation: Documentation should be written surrounding best practices for Shaman contracts. If a Shaman contract is upgradeable, the DAO should consider only allowing Baal to trigger upgrades. If a Shaman contract is a Governor, it will have the power to change the Trusted Forwarder and impersonate any `msg.sender`.

Update: The team has addressed this in their documentation.

QSP-13 Upgradability

Severity: *Informational*

Status: Acknowledged

File(s) affected: `Baal.sol`, `LootERC20`, `SharesERC20`

Description: Many contracts within the project are upgradeable. While this is not a vulnerability, users should be aware that the behavior of the contracts could drastically change if the contracts are upgraded. Furthermore, new vulnerabilities not present during the audit could be introduced in upgraded versions of the contract, or if contract upgrade deployments are not done correctly.

Recommendation: The contract's upgradeability and any reasons for future upgrades should be communicated to users beforehand.

Update: The team has addressed this in their documentation.

QSP-14 `msg.sender` Can Be Overridden.

Severity: *Informational*

Status: Acknowledged

File(s) affected: `Baal.sol`

Description: `Baal` implements the `BaseRelayRecipient` so that it can support Meta transactions. This comes with some security considerations, as outlined in EIP-2771:

> A bad forwarder may allow forgery of the `msg.sender` returned from `_msgSender()` and allow transactions to appear to be coming from any address. This means a recipient contract should be very careful which forwarder it trusts and whether this can be modified. The power to change the forwarder trusted by a recipient is equivalent to giving full control over the contract. If this kind of control over the recipient is acceptable, it is recommended that only the owner of the recipient contract be able to modify which forwarder is trusted. Otherwise best to leave it unmodifiable.

Recommendation: Make sure only trusted addresses are trusted as forwarders. Make the power of a Governor Shaman changing the forwarder clear to users.

Update: The team has addressed this in their documentation.

QSP-15 External Calls to Malicious Contracts

Severity: *Undetermined*

Status: Acknowledged

File(s) affected: `Baal.sol`

Description: When a user calls `Baal.ragequit()`, ideally they burn their shares and loot in exchange for token's belonging to the DAO's treasury, however the user provides the token addresses for the tokens they want to be paid out in. Because there is no validation done on the addresses provided for the tokens, the user can pass in addresses belonging to malicious contracts, which may not behave as expected.

Recommendation: While no immediate threats were determined due to the heavy use of re-entrancy guards, we recommend validating the tokens being provided for ragequit to avoid any potential exploits.

Update: The team has addressed this in their documentation.

QSP-16 Proposals Can Pass without a Valid Sponsor

Severity: *Undetermined*

Status: Acknowledged

File(s) affected: `Baal.sol`

Description: In `Baal`, a proposal needs a valid sponsor for voting to commence. A sponsor is considered valid if their balance of shares meets the `sponsorThreshold`. If their balance falls below the `sponsorThreshold`, anyone can call the `cancelProposal()` function to cancel the proposal. However, this is not guaranteed to happen. If no one chooses to call `cancelProposal()`, it is possible for the proposal to proceed through the proposal flow as if it had a valid sponsor.

Recommendation: Please clarify whether it is permissible for a proposal to pass without a valid sponsor. If not, update `processProposal()` to confirm that the proposal's sponsor is still valid.

Update: The team has addressed this in their documentation.

Automated Analyses

Slither

```
Contract locking ether found:
    Contract BaalSummoner (contracts/BaalSummoner.sol#10-318) has payable functions:
     - BaalSummoner.summonBaalFromReferrer(bytes,bytes[],uint256,bool,bytes32) (contracts/BaalSummoner.sol#120-145)
    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 Baal.processProposal(uint32,bytes) (contracts/Baal.sol#472-530):
    External calls:
    - success = processActionProposal(proposalData) (contracts/Baal.sol#523)
        - IGuard(guard).checkTransaction(to,value,data,operation,0,0,0,address(0),address(0),bytes(0x),msg.sender) (node_modules/@gnosis.pm/zodiac/contracts/core/Module.sol#51-65)
        - success = IAvatar(target).execTransactionFromModule(to,value,data,operation) (node_modules/@gnosis.pm/zodiac/contracts/core/Module.sol#67-72)
        - IGuard(guard).checkAfterExecution(bytes32(0x),success) (node_modules/@gnosis.pm/zodiac/contracts/core/Module.sol#74)
    State variables written after the call(s):
    - prop.status[3] = true (contracts/Baal.sol#525)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

Baal.setUp(bytes)._avatar (contracts/Baal.sol#235) lacks a zero-check on :
        - avatar = _avatar (contracts/Baal.sol#247)
        - target = _avatar (contracts/Baal.sol#248)
Baal.setUp(bytes)._multisendLibrary (contracts/Baal.sol#234) lacks a zero-check on :
        - multisendLibrary = _multisendLibrary (contracts/Baal.sol#257)
Baal.executeAsBaal(address,uint256,bytes)._to (contracts/Baal.sol#571) lacks a zero-check on :
        - (success) = _to.call{value: _value}(_data) (contracts/Baal.sol#575)
BaalSummoner.constructor(address,address,address,address,address,address,address,address)._template (contracts/BaalSummoner.sol#48) lacks a zero-check on :
        - template = _template (contracts/BaalSummoner.sol#60)
BaalSummoner.constructor(address,address,address,address,address,address,address,address)._gnosisFallbackLibrary (contracts/BaalSummoner.sol#50) lacks a zero-check on :
        - gnosisFallbackLibrary = _gnosisFallbackLibrary (contracts/BaalSummoner.sol#62)
BaalSummoner.constructor(address,address,address,address,address,address,address,address)._gnosisMultisendLibrary (contracts/BaalSummoner.sol#51) lacks a zero-check on :
        - gnosisMultisendLibrary = _gnosisMultisendLibrary (contracts/BaalSummoner.sol#63)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Reentrancy in Baal.submitProposal(bytes,uint32,uint256,string) (contracts/Baal.sol#298-360):
    External calls:
    - (_success) = target.call{value: msg.value}() (contracts/Baal.sol#315)
    State variables written after the call(s):
    - latestSponsoredProposalId = proposalCount (contracts/Baal.sol#344)
    - proposalCount ++ (contracts/Baal.sol#322)
    - proposals[proposalCount] = Proposal(proposalCount,latestSponsoredProposalId,uint32(block.timestamp),uint32(block.timestamp) + votingPeriod,uint32(block.timestamp) + votingPeriod + gracePeriod,expiration,baalGas,0,0,0,
(false,false,false,false),_msgSender(),proposalDataHash,details) (contracts/Baal.sol#323-340)
    - proposals[proposalCount] = Proposal(proposalCount,0,0,0,0,expiration,baalGas,0,0,0,(false,false,false,false),address(0),proposalDataHash,details) (contracts/Baal.sol#323-340)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
```

Code Documentation

Overall the code is well documented and makes use of NatSpec. The HausDAO team has excellent documentation outlining the contracts and how Moloch V3 builds on its predecessors.

Adherence to Best Practices

- In the `Baal` contract, replace `draft-EIP712` with `EIP712`.
- `Executor` and `GnosisSafe` are imported in `Baal` but not used.
- Proposal id is `uint32` but defined as `uint256` in `proposals`.
- `0x70a08231` in `Baal.sol#L628` can be defined as a constant.
- It is better to define an initialize function for `BaalVotes` contract and call `__ERC20Permit_init` in that function.
- Define an enum for shamans
- `BaalSummoner` inherits from `ModuleProxyFactory`, so there is no need to construct `BaalSummoner` with `moduleProxyFactory`, since the calls to `ModuleProxyFactory.deployModule()#L230,280` can be made by `BaalSummoner`.
- If `Baal.proposalOffering` is equal to zero, avoid calling `target.call{value: msg.value}` in `submitProposal()#L315`.
- `Baal.sol`: Set the constructor to call `_disableInitializers()` rather than use the `initializer` modifier.
- `Baal.sol`: Consider moving the checks for proposal expiration, reaching quorum, and meeting the minimum retention percentage from `processProposal()` to `state()` in the `ProposalState.Defeated` section.
- `Baal.sol`: Update the function casing to match its visibility: `processActionProposal()`.
- `Baal.sol`: In `getProposalStatus()`, require that the proposal exists.

- `BaalSummoner.sol`: `BaalSummoner` implements `ModuleProxyFactory` but makes the calls using its `moduleProxyFactory` variable. Either remove `ModuleProxyFactory` as a parent or make the calls through `BaalSummoner` rather than the `moduleProxyFactory` variable.
- `BaalSummoner.sol`: Update the function casing to match its visibility: `deployAndSetupSafe()`.

Test Results

Test Suite Results

The test suite was run by calling `npx hardhat test`.

```
Compiled 81 Solidity files successfully

  Baal contract
    constructor
       ✓ verify deployment parameters
    token ownership
       ✓ can not transfer ownership when not owner
       ✓ can not be upgraded when not owner
       ✓ can renounce loot token ownership
       ✓ can renounce shares token ownership
       ✓ can change shares token ownership to avatar
       ✓ can change loot token ownership to avatar
upgrade
       ✓ can eject and upgrade token with eoa
    shaman actions - permission level 7 (full)
       ✓ setAdminConfig
       ✓ mint shares - recipient has shares
       ✓ mint shares - new recipient
       ✓ mint shares - recipient has delegate - new shares are also delegated
       ✓ mint shares - zero mint amount - no votes
       ✓ mint shares - require fail - array parity
       ✓ burn shares
       ✓ burn shares - require fail - array parity
       ✓ burn shares - require fail - insufficent shares
       ✓ mint loot
       ✓ mint loot - require fail - array parity
       ✓ burn loot
       ✓ burn loot - require fail - array parity
       ✓ burn loot - require fail - insufficent shares
       ✓ set trusted forwarder
       ✓ have shaman mint and burn _delegated_ shares
       ✓ setGovernanceConfig
       ✓ setGovernanceConfig - doesnt set voting/grace if =0
       ✓ cancelProposal - happy case - as gov shaman
       ✓ cancelProposal - happy case - as proposal sponsor
       ✓ cancelProposal - happy case - after undelegation
       ✓ cancelProposal - require fail - not cancellable by rando
       ✓ cancelProposal - require fail - !voting (submitted)
       ✓ cancelProposal - require fail - !voting (grace)
       ✓ cancelProposal - require fail - !voting (defeated)
       ✓ cancelProposal - require fail - !voting (cancelled)
       ✓ cancelProposal - require fail - !voting (ready)
       ✓ cancelProposal - require fail - !voting (processed)
    shaman permissions: 0-6
       ✓ permission = 0 - all actions fail
       ✓ permission = 1 - admin actions succeed
       ✓ permission = 2 - manager actions succeed
       ✓ permission = 3 - admin + manager actions succeed
       ✓ permission = 4 - governor actions succeed
       ✓ permission = 5 - admin + governor actions succeed
       ✓ permission = 6 - manager + governor actions succeed
    shaman locks
       ✓ lockAdmin
       ✓ lockManager
       ✓ lockGovernor
    setShamans - adminLock (1, 3, 5, 7)
       ✓ setShamans - 0 - success
       ✓ setShamans - 1 - fail
       ✓ setShamans - 2 - success
       ✓ setShamans - 3 - fail
       ✓ setShamans - 4 - success
       ✓ setShamans - 5 - fail
       ✓ setShamans - 6 - success
       ✓ setShamans - 7 - fail
    setShamans - managerLock (2, 3, 6, 7)
       ✓ setShamans - 0 - success
       ✓ setShamans - 1 - success
       ✓ setShamans - 2 - fail
       ✓ setShamans - 3 - fail
       ✓ setShamans - 4 - success
       ✓ setShamans - 5 - success
       ✓ setShamans - 6 - fail
       ✓ setShamans - 7 - fail
    setShamans - governorLock (4, 5, 6, 7)
       ✓ setShamans - 0 - success
       ✓ setShamans - 1 - success
       ✓ setShamans - 2 - success
       ✓ setShamans - 3 - success
       ✓ setShamans - 4 - fail
       ✓ setShamans - 5 - fail
       ✓ setShamans - 6 - fail
       ✓ setShamans - 7 - fail
    setShamans - all locked
       ✓ setShamans - 0 - success
       ✓ setShamans - 1 - fail
       ✓ setShamans - 2 - fail
       ✓ setShamans - 3 - fail
       ✓ setShamans - 4 - fail
       ✓ setShamans - 5 - fail
       ✓ setShamans - 6 - fail
       ✓ setShamans - 7 - fail
    erc20 shares - approve
       ✓ happy case
       ✓ overwrites previous value
    erc20 shares - transfer
       ✓ transfer to first time recipient - auto self delegates
       ✓ require fails - shares paused
       ✓ require fails - insufficient balance
       ✓ 0 transfer - doesnt update delegates
       ✓ self transfer - doesnt update delegates
       ✓ transferring to shareholder w/ delegate assigns votes to delegate
    erc20 shares - transferFrom
       ✓ transfer to first time recipient
       ✓ require fails - shares paused
       ✓ require fails - insufficeint approval
    erc20 loot - approve
       ✓ happy case
       ✓ overwrites previous value
    erc20 loot - transfer
       ✓ sends tokens, not votes
       ✓ require fails - loot paused
       ✓ require fails - insufficient balance
    erc20 loot - transferFrom
       ✓ sends tokens, not votes
       ✓ require fails - loot paused
       ✓ require fails - insufficient balance
       ✓ require fails - insufficeint approval
    submitProposal
```

```
    ✓ happy case
    ✓ require fail - expiration passed
    ✓ edge case - expiration exists, but far enough ahead
  sponsorProposal
    ✓ happy case
    ✓ require fail - proposal expired
    ✓ edge case - expiration exists, but far enough ahead 2
    ✓ require fail - not sponsor
    ✓ edge case - just enough shares to sponsor
    ✓ require fail - proposal doesnt exist
    ✓ require fail - already sponsored
  submitVote (w/ auto self-sponsor)
    ✓ happy case - yes vote
    ✓ happy case - no vote
    ✓ require fail - voting period has ended
    ✓ require fail - already voted
    ✓ require fail - not a member
    ✓ scenario - two yes votes
  submitVote (no self-sponsor)
    ✓ require fail - voting not started
    ✓ scenario - increase shares during voting
    ✓ scenario - decrease shares during voting
  submitVoteWithSig (w/ auto self-sponsor)
    ✓ happy case - yes vote
    ✓ fail case - fails with different voter
    ✓ fail case - cant vote twice
  delegateBySig
    ✓ happy case
    ✓ require fail - nonce is re-used
    ✓ require fail - signature expired
  processProposal
    ✓ happy case yes wins
    ✓ require fail - not enough gas
    ✓ has enough baalGas
    ✓ require fail - no wins, proposal is defeated
    ✓ require fail - proposal does not exist
    ✓ require fail - no sponser
    ✓ require fail - prev proposal not processed
    ✓ require fail - proposal data mismatch on processing
    ✓ require fail - proposal not in voting
    ✓ require fail - proposal cancelled
    ✓ require fail - proposal expired 2
    ✓ edge case - exactly at quorum
    ✓ edge case - just under quorum
    ✓ edge case - exactly at minRetentionPercent
    ✓ edge case - just below minRetentionPercent - shares+loot
    ✓ edge case - just below minRetentionPercent - just shares
    ✓ edge case - just below minRetentionPercent - just loot
    ✓ scenario - offer tribute unsafe
    ✓ scenario - two propsals, prev is processed
    ✓ scenario - two propsals, prev is defeated
    ✓ scenario - two propsals, prev is cancelled
    ✓ happy case - mint shares via proposal
    ✓ happy case - burn shares via proposal
    ✓ happy case - mint loot via proposal
    ✓ happy case - burn loot via proposal
  ragequit
    ✓ happy case - full ragequit
    ✓ happy case - partial ragequit
    ✓ happy case - full ragequit to different address
    ✓ happy case - full ragequit - two tokens
  ragequit
    ✓ collects tokens not on the list
    ✓ require fail - enforces ascending order
    ✓ require fail - prevents actual duplicate
  getCurrentVotes
    ✓ happy case - account with votes
    ✓ happy case - account without votes
  getPriorVotes
    ✓ happy case - yes vote
    ✓ happy case - no vote
    ✓ require fail - timestamp not determined

Baal contract - offering required
  submitProposal
    ✓ submit proposal
    ✓ happy case - sponsors can submit without offering, auto-sponsors
    ✓ edge case - sponsors can submit without offering at threshold
    ✓ require fail - no offering offered

Baal contract - summon baal with current safe
  Baal summoned after safe
    ✓ should have the expected address of the module the same as the deployed

Loot ERC20 contract
  constructor
    ✓ creates an unusable template
    ✓ require fail - initializer (setup) cant be called twice on loot
    ✓ require fail - initializer (setup) cant be called on singleton
  er20 loot - authorized minting, burning
    ✓ happy case - allows baal to mint when loot not paused
    ✓ happy case - allows baal to mint when loot paused
    ✓ require fail - non baal tries to mint
    ✓ happy case - allows baal to burn when loot not paused
    ✓ happy case - allows baal to burn when loot paused
    ✓ require fail - non baal tries to burn
    ✓ require fail - non baal tries to send to 0
  er20 loot - restrict transfer
    ✓ happy case - allows loot to be transferred when enabled
    ✓ require fail - tries to transfer loot when paused
    ✓ happy case - allows loot to be transfered with approval when enabled
    ✓ require fail - tries to transfer with approval loot when paused
  erc20 loot - increase allowance with permit
    ✓ happy case - increase allowance with valid permit
    ✓ Require fail -  invalid nonce
    ✓ Require fail - invalid chain Id
    ✓ Require fail - invalid name
    ✓ Require fail - invalid address
    ✓ Require fail - invalid owner
    ✓ Require fail - invalid spender
    ✓ Require fail - invalid amount
    ✓ Require fail - invalid deadline
    ✓ Require fail - expired deadline

  Tribute proposal type
    Dangerous proposal tribute
      ✓ Allows applicant to tribute tokens in exchagne for shares
      ✓ EXPLOIT - Allows another proposal to spend tokens intended for tribute
    Baal with NO proposal offering - Safe Tribute Proposal
{ state: 6, propStatus: [ false, true, true, false ] }
      ✓ allows external tribute minion to submit share proposal in exchange for tokens
{ state: 6, propStatus: [ false, true, true, false ] }
      ✓ tribute without proposal offering
    Baal with proposal offering - Safe Tribute Proposal
{ state: 6, propStatus: [ false, true, true, false ] }
      ✓ allows external tribute minion to submit share proposal in exchange for tokens
{ state: 6, propStatus: [ false, true, true, false ] }
      ✓ should not fail to tribute without offering
      ✓ fails to tribute without offering

·---------------------------------------------|---------------------------|-------------|-----------------------------·
|             Solc version: 0.8.7             ·  Optimizer enabled: true  ·  Runs: 100  ·  Block limit: 30000000 gas  |
··············|··············|··············|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|
|  Methods                                                                                                            |
··············|··············|··············|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|
|  Contract    ·  Method      ·  Min         ·  Max  ·  Avg  ·  # calls  ·  eur (avg)  |
··············|··············|··············|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|·····|
|  Baal        ·  burnLoot    ·      60614   ·  65439  ·  61828  ·  4  ·  -  |
```

```
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  burnShares                 ·      99178  ·     123903  ·     109074  ·            5  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  cancelProposal             ·      78286  ·     101048  ·      95700  ·            9  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  mintLoot                   ·      65198  ·      82293  ·      78012  ·            4  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  mintShares                 ·      59465  ·     169123  ·     140882  ·           18  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  processProposal            ·     101887  ·     322863  ·     191069  ·          114  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  ragequit                   ·      89266  ·     197575  ·     163177  ·           10  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  setAdminConfig             ·      82179  ·     107955  ·      95072  ·            8  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  setGovernanceConfig        ·      77466  ·     122854  ·      97791  ·           11  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  setTrustedForwarder        ·      31584  ·      31610  ·      31599  ·            4  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  sponsorProposal            ·     101538  ·     107906  ·     106632  ·            5  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  submitProposal             ·     164563  ·     269779  ·     233425  ·          158  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  submitVote                 ·     119916  ·     158852  ·     154999  ·          128  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal              ·  submitVoteWithSig          ·     195329  ·     195337  ·     195333  ·            2  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  BaalSummoner      ·  summonBaal                 ·    1491576  ·    1491588  ·    1491586  ·           62  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  BaalSummoner      ·  summonBaalAndSafe          ·    1728329  ·    1899963  ·    1738204  ·          348  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  ERC20Upgradeable  ·  approve                    ·      34012  ·      51124  ·      48541  ·           15  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  ERC20Upgradeable  ·  transfer                   ·      38954  ·     211116  ·     128576  ·           12  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  ERC20Upgradeable  ·  transferFrom               ·      66670  ·     214223  ·     116709  ·            3  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Loot              ·  mint                       ·          -  ·          -  ·     102637  ·            1  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Loot              ·  permit                     ·          -  ·          -  ·      81977  ·            1  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  MockBaal          ·  burnLoot                   ·          -  ·          -  ·      52050  ·            2  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  MockBaal          ·  mintLoot                   ·      68875  ·      85987  ·      83542  ·           28  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  MockBaal          ·  setLootPaused              ·          -  ·          -  ·      59322  ·            4  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Shares            ·  delegate                   ·     129408  ·     141694  ·     134320  ·            5  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Shares            ·  delegateBySig              ·          -  ·          -  ·     177473  ·            2  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  TestAvatar        ·  enableModule               ·      43917  ·      43929  ·      43928  ·           31  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  TestERC20         ·  approve                    ·      46059  ·      46083  ·      46077  ·            8  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  TestERC20         ·  transfer                   ·      51401  ·      51413  ·      51406  ·           16  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  TributeMinion     ·  submitTributeProposal      ·     295445  ·     302660  ·     300537  ·            4  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Deployments                                     ·                                       ·  % of limit   ·             |
·····················|·····························|·············|·············|·············|···············|·············
|  Baal                                            ·          -  ·          -  ·    5345285  ·       17.8 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  BaalLessShares                                  ·          -  ·          -  ·    2262192  ·        7.5 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  BaalSummoner                                    ·    2087288  ·    2087324  ·    2087320  ·          7 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  CompatibilityFallbackHandler                    ·          -  ·          -  ·     874768  ·        2.9 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  GnosisSafe                                      ·          -  ·          -  ·    2706330  ·          9 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  GnosisSafeProxyFactory                          ·          -  ·          -  ·     608497  ·          2 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Loot                                            ·          -  ·          -  ·    2302723  ·        7.7 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  MockBaal                                        ·          -  ·          -  ·     557766  ·        1.9 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  ModuleProxyFactory                              ·          -  ·          -  ·     257108  ·        0.9 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  MultiSend                                       ·          -  ·          -  ·     181745  ·        0.6 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Poster                                          ·          -  ·          -  ·     153433  ·        0.5 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  Shares                                          ·          -  ·          -  ·    2569949  ·        8.6 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  TestAvatar                                      ·          -  ·          -  ·     452713  ·        1.5 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  TestERC20                                       ·     489412  ·     489436  ·     489422  ·        1.6 %  ·          -  |
·····················|·····························|·············|·············|·············|···············|·············
|  TributeMinion                                   ·          -  ·          -  ·     883637  ·        2.9 %  ·          -  |
·----------------------|-----------------------------|-------------|-------------|-------------|---------------|-------------·

  196 passing (1m)
```

Code Coverage

The code coverage was gathered by running `npx hardhat coverage --network localhost`.

```
-----------------------|----------|----------|----------|----------|----------------|
File                   |  % Stmts | % Branch |  % Funcs |  % Lines |Uncovered Lines |
-----------------------|----------|----------|----------|----------|----------------|
 contracts/            |    95.32 |    82.56 |    91.55 |    94.53 |                |
    Baal.sol           |    96.62 |    88.73 |    93.33 |    95.76 |... 9,1016,1046 |
    BaalSummoner.sol   |    90.63 |    43.75 |    88.89 |    90.32 |... 139,146,147 |
    LootERC20.sol      |    92.86 |    66.67 |    77.78 |    92.86 |             42 |
    SharesERC20.sol    |      100 |     62.5 |      100 |      100 |                |
 contracts/fixtures/   |      100 |      100 |      100 |      100 |                |
    GnosisImports.sol  |      100 |      100 |      100 |      100 |                |
 contracts/interfaces/ |      100 |      100 |      100 |      100 |                |
    IAdminShaman.sol   |      100 |      100 |      100 |      100 |                |
    IBaal.sol          |      100 |      100 |      100 |      100 |                |
    IBaalToken.sol     |      100 |      100 |      100 |      100 |                |
    IGovernorShaman.sol|      100 |      100 |      100 |      100 |                |
    IManagerShaman.sol |      100 |      100 |      100 |      100 |                |
 contracts/mock/       |       70 |    27.78 |       70 |    75.56 |                |
    BaalLessToken.sol  |    85.71 |       50 |    66.67 |    85.71 |             51 |
    MockBaal.sol       |    77.78 |       25 |      100 |     87.5 |             29 |
    TestAvatar.sol     |       25 |       20 |    33.33 |       25 |... 50,60,62,63 |
    TestERC20.sol      |      100 |       50 |      100 |      100 |                |
 contracts/tools/      |      100 |       60 |      100 |      100 |                |
    TributeMinion.sol  |      100 |       60 |      100 |      100 |                |
 contracts/utils/      |    92.73 |       80 |      100 |       98 |                |
    BaalVotes.sol      |    92.59 |       80 |      100 |    97.96 |            169 |
    Poster.sol         |      100 |      100 |      100 |      100 |                |
-----------------------|----------|----------|----------|----------|----------------|
All files              |     92.4 |    76.96 |    88.46 |    93.18 |                |
-----------------------|----------|----------|----------|----------|----------------|
```

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

```
48584183d5dfbb41b1ed3fb743985cc53923bd9f6af9989e1827575a1e9cc41e ./contracts/Baal.sol
b1aa92ba3f9a3933d827d22c292a7118e13fb646bd8ad3c2e0b69923e2f106be ./contracts/SharesERC20.sol
4a1819ff6189d67894746289f939a9b5abc0ef69f0157c7f775779a07cc8563c ./contracts/BaalSummoner.sol
13bf3956d21633a404ae7cd47cfb52f36b1c3bfdd8253444bd55783920f8f1c9 ./contracts/LootERC20.sol
e3d5ef55049261a632e6e9574cb2445a916d890917bf406d6dd2d8ac87a00e13 ./contracts/interfaces/IAdminShaman.sol
111d36e7e8ef1159593ab27fc3d2b4064800a6a58d0f3e92eb8ee786e039ba80 ./contracts/interfaces/IBaalToken.sol
9248de2179539a93e05b4cdf71a4872b254b3a45e31ca142279d3c6cc5a7ed65 ./contracts/interfaces/IGovernorShaman.sol
4cca2b0d34eea28c4eb5e12e88f61ce38c91c0febcdba7bf89c5f4c2f1c0ab2e ./contracts/interfaces/IManagerShaman.sol
ff98f47241057a547099b6ecc8e570f8c68e43a808fa12553a082389e65c21d5 ./contracts/interfaces/IBaal.sol
ebbdcb6261efc30da5643e118d2c47c1b9fd6c5355ef478c87da0cbd901d8835 ./contracts/fixtures/GnosisImports.sol
42efee9b4eb7a1e2460d723e9faa15d8a9a31e13d152af80c0236cb0f2623d47 ./contracts/utils/Poster.sol
1056b977add10aad3f5db765c2c48187dc15fad6928f7f26f7c4dd12276c4081 ./contracts/utils/BaalVotes.sol
2ec75e440c0839fe30d481a82e4012fdb4ccfcf0d40bf1d08f33a333c905b88f ./contracts/mock/MockBaal.sol
1e36637498da3d4d6a746beed29fb27d5b9e2c32e2a9b740e68e2f78c2726f77 ./contracts/mock/BaalLessToken.sol
5d4de6212c473336f189a1b32c297ac149d3ba57f6a71e522da037e6231ef3f4 ./contracts/mock/TestERC20.sol
aab276f15edc4522d77218d1c5fdceac577fc3266cc36955e076357089e389f6 ./contracts/mock/TestAvatar.sol
c3ff47fe7d9eab05f5204858986ede5e825eed398a77b2fefc9939a70b0f3ca7 ./contracts/tools/TributeMinion.sol
```

Tests

```
76c5535497564bb1211c30b193cd736834f5b2f770957359e2703d78283115e8 ./test/BaalSafe.test.ts
460ec505d479a9de8c7783c2a810f9a0b0bd9e034c3b12488a387b84eba53643 ./test/Tribute.test.ts
9dc2607baa6e3665babe683b7721749902a750afcef6a143b28dd180aad8c475 ./test/LootERC20.test.ts
```

Changelog

- 2022-09-21 - Initial report

About Quantstamp

Quantstamp is a global leader in blockchain security backed by Pantera, Softbank, and Commonwealth among other preeminent investors. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its white glove security and risk assessment services.

The team consists of web3 thought leaders hailing from top organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Many of the auditors hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 250 audits and secured over $200 billion in digital asset risk from hackers. In addition to providing an array of security services, Quantstamp facilitates the adoption of blockchain technology through strategic investments within the ecosystem and acting as a trusted advisor to help projects scale.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Aave, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT