Quantstamp

Hypha Liquid Staking

Executive Summary

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

Type	Liquid Staking Protocol
Timeline	2025-07-21 through 2025-07-31
Language	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	Audit Notion Page
Source Code	• https://github.com/multisig-labs/gogopool #11e7f98
Auditors	• Ruben Koch Senior Auditing Engineer • Andrei Stefan Auditing Engineer • Hamed Mohammadi Auditing Engineer

Documentation quality	High
Test quality	High
Total Findings	11 Fixed: 8 Acknowledged: 3
High severity findings	1 Fixed: 1
Medium severity findings	0
Low severity findings	9 Fixed: 6 Acknowledged: 3
Undetermined severity findings	0
Informational findings	1 Fixed: 1

Summary of Findings

In this audit, we reviewed an extension and upgrade to the Hypha protocol. The upgrade component we reviewed revolves around Hypha's liquid staking token, called `ggAVAX` (to be renamed to `stAVAX` soon). The upgrade essentially introduces a withdrawal

Users can deposit AVAX into the `ggAVAX` ERC-4626 vault, receiving `ggAVAX` in return. Trusted entities can withdraw the AVAX from the vault to use it as the bond to participate in AVAX's Proof of Stake protocol, accruing staking rewards. These entities are then supposed to streamline accumulated rewards back to the vault, as well as return the initial base amount to the vault if the minipool is teared down. The main upgrade to the `ggAVAX` introduces a withdrawal delay between the redemption of `ggAVAX` back to `AVAX`. This gives the protocol maintainers increased reaction time to possibly tear down existing minipool nodes and returning the `AVAX` to the vault, in case incoming withdrawal requests exceed the current amount of `AVAX` held directly by the vault. This is achieved with the newly added `WithdrawQueue` contract.

Additionally, a `pstAVAX` token is introduced that essentially provides an additional yield farming strategy. Depositors forfeit their staking rewards, which are funnelled to `ggAVAX` holders, in favour of participating in an out of scope, off-chain point farming program.

Overall the code is well written and the test suite is robust. One high severity issue and some low severity issues have been identified, mainly revolving around edge cases, as well as numerous suggestions to adhere to coding best practices.

Fix-Review Update 2025-08-19:
All issues identified in the report have been fully fixed or reasonably acknowledged. Furthermore, additional tests have been added to comprehensively validate some of the fixes, expanding the already robust test suite further.

ID	DESCRIPTION	SEVERITY	STATUS
HYP-1	Potentially Stale Prices Due To Unstripped Yield Can Cause Dilution	● High	Fixed
HYP-2	Possibility of Multiple Default Admin Entities and Incorrect Admin Transfer	● Low	Fixed
HYP-3	Potential Reverts Due to Underflows in `depositFromStaking()`	● Low	Fixed

ID	DESCRIPTION	SEVERITY	STATUS
HYP-4	**Consider Adding Rate Limiting Mechanisms to Protect Against Rogue Trusted Entities**	● Low ⓘ	Acknowledged
HYP-5	**Risk of Duplicate Initialization**	● Low ⓘ	Acknowledged
HYP-6	**Minor Potential DoS vector on** `requestsByOwner`	● Low ⓘ	Fixed
HYP-7	**Assets Intended for ggAvax Holders Repurposed for Fulfilling Withdrawal Requests in Certain Cases**	● Low ⓘ	Fixed
HYP-8	**Inconsistent Application of Sanity Checks on** `withdrawForStaking(s)` **Functions**	● Low ⓘ	Acknowledged
HYP-9	`depositFromStaking()` **Can Revert in Certain States**	● Low ⓘ	Fixed
HYP-10	**Potential Reverts of the** `getExcessShares()` **function due to Underflow**	● Low ⓘ	Fixed
HYP-11	**Improvements to** `try-catch` **Logic in** `depositFromStaking`	● Informational ⓘ	Fixed

Assessment Breakdown

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

Disclaimer
Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

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

1. Code review that includes the following
   1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
   2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   3. 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:
   1. 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.
   2. 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, clarity, 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.

Scope

Files Included

Repo: https://github.com/multisig-labs/gogopool(11e7f984d6025d93cadc3f28cba7665d89fe6d06)

Files: https://github.com/multisig-labs/gogopool/pull/5

- contracts/contract/tokens/TokenpstAVAX.sol
- contracts/contract/WithdrawQueue.sol
- contracts/contract/tokens/TokenggAVAX.sol
- script/upgrade-liquid-staking-system.s.sol

Repo: `https://github.com/multisig-labs/gogopool`

Operational Considerations

- All of the three contracts are upgradeable. We assume that all future protocol upgrades go through a planned, audited upgrade process.
- We consider that all trusted roles in the system behave honestly and as expected
- The `ggAVAX` vault can be paused, which however mainly stops additional deposits from entering the protocol.

Key Actors And Their Capabilities

The `ggAVAX` vault defines roles for trusted stakers and minipool managers that are able to withdraw AVAX from the vault in order to use those assets as a bond in order to generate staking rewards. These roles are expected to streamline the rewards back in to the vault, as well as return the base assets once the minipool node is teared down. Furthermore, there is an admin role, intended to be held by a single address that can assign these roles.

The `pstAVAX` contract defines an `owner` address that can pause the contract that can stop all protocol interactions, including deposits and withdrawals from the `pstAVAX` vault. It also provides a recovery function for non-protocol-related ERC-20 tokens.

The `WithdrawQueue` contract defines a admin role that can set numerous maintenance parameters, most importantly including the withdrawal delay period and the expiration date for the claim window of fulfilled withdrawal requests. Withdrawal requests get fulfilled by holders of the depositor role, which essentially allocate `AVAX` to currently pending withdrawal requests in the queue.

Findings

HYP-1

Potentially Stale Prices Due To Unstripped Yield Can Cause Dilution

● High ⓘ Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `a3b45138cbffb470c637e0486dcb606a7a32d27e`.
The client provided the following explanation:

```
Followed the suggested remediation and this issue was fixed by calling TokenpstAVAX::stripYield anytime
a user deposits or withdraws from TokenpstAVAX. This ensures that all users get an up to date exchange
rate via burning excess vault shares in pstAVAX.
```

We do want to note that periodically calling `stripYield()` will be necessary in order to make a similar attack economically unfeasible for the `TokenggAvax` contract.

File(s) affected: `contracts/contract/tokens/TokenpstAVAX.sol`, `contracts/contract/WithdrawQueue.sol`, `contracts/contract/tokens/TokenggAVAX.sol`

Description: The `pstAVAX` vault is supposed to only hold as many `ggAVAX` tokens as are needed to cover withdrawals of the `AVAX` pegged `pstAVAX` supply. As `ggAVAX` share tokens are expected to increase in value compared to `(pst)AVAX` due to accruing yield, the `ggAVAX` the vault needs to hold can over time be reduced. This is done with the `stripYield()` function that is expected to be called periodically, internally relying on a formula defined in the `getExcessShares()` function. The `stripYield()` function burns the excessive `ggAVAX`, which serves as additional value accrual for `ggAVAX` token holders, as it causes the exchange rate between `AVAX` and `ggAVAX` to increase.

However, this reserve of to-be-burned `ggAVAX` in the `pstAVAX` vault causes the `ggAVAX` exchange rate to be stale, as as soon as the burn is materialized, the exchange rate will immediately increase due to the reduced share token supply. Non-zero stripped yield in the `pstAVAX` vault causes any `AVAX` <> `ggAVAX` conversion to be inaccurate, causing `(w)AVAX` deposits to receive more `ggAVAX` than expected and `ggAVAX` withdraws (ignoring the queue delay) to receive less `AVAX` than expected.

This also causes problems between `pstAVAX` <> `ggAVAX`, as the `pstAVAX` vault ends up minting more `ggAVAX` than it should, when it internally deposits into `ggAVAX` during the `pstAVAX` minting, which immediately causes more excessive shares being accrued in the vault.

While the excessive shares in the `pstAVAX` are to be burned at some point, direct deposits into the `ggAVAX` vault cause excessive share token minting, causing value accrual to be dilluted to the new depositor.

This also opens up an attack vector where users can deposit into the `ggAVAX` contract, receive more `ggAVAX` than expected due to the stale exchange rate, call `pstAVAX.stripYield()` to cause the exchange rate to increase and then immediately withdraw again, receiving back more `AVAX` than on the initial deposit, essentially due to stealing some of the value accrual from the burning. Granted, the withdraw delay applies here. However, to a slightly lesser extent this attack also applies to `pstAVAX` deposit, without any withdrawal delays, though here the received `pstAVAX` amount is pegged.

Recommendation:

- Make `stripYield()` public and remove the `nonReentrant` modifier. Furthermore, make it gracefully return `0` instead of reverting if `getExcessShares()` returns 0.
- In the pstAVAX contract, basically before any implicit or explicit conversion, a preliminary call `pstAVAX.stripYield()` should be made. Namely in the `withdraw()`, withdrawViaQueue() and `_deposit()` function.
- In the `ggAVAX` contract, too, before any implicit or explicit conversion, `pstAVAX.stripYield()` should be called.

HYP-2 Possibility of Multiple Default Admin Entities and Incorrect Admin Transfer

• Low ⓘ Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `b35cb8acaa727a1d0df119b19dbe476e1ec63f54`.
The client provided the following explanation:

```
We followed the suggested remediation and now revert calls to TokenggAVAX::grantRole if the role that's
granted is the DEFAULT_ADMIN_ROLE. We also removed the TokenggAVAX::renounceAdmin function to ensure
that the pendingAdmin always finalizes the admin transfer
```

File(s) affected: `contracts/contract/tokens/TokenggAVAX.sol`

Description: The upgrade of the `TokenggAVAX` contract introduces access control heavily inspired by OpenZeppelin's `AccessControl` library that involves a default admin role, but state variable management for a pending and current admin that essentially enables a two-step transfer of the role. Two minor issues have been identified:

1. The two state variables can be fetched via `admin()` and `pendingAdmin()`, implying that there can only be one default admin. However, the default admin can call the `grantRole()` function, assigning the `DEFAULT_ADMIN_ROLE` to another account, which also overrides the state variable, but the `msg.sender`'s `DEFAULT_ADMIN_ROLE` remains un-revoked, effectively leaving the contract with two default admins, yet only one being tracked via `admin()`.
2. It is our opinion that the 2-step transfer for the admin role should be solely completable by the `_pendingAdmin` via the `acceptAdmin()` function, essentially making sure that the `_pendingAdmin` address is indeed a controlled address. However, additionally, the current admin can also call `renounceAdmin()`, which will remove the caller itself as an admin, as long as there is any non-zero `_pendingAdmin`, making this process fully one-sided.

Recommendation:

1. In `grantRole()`, if the `role` parameter is `DEFAULT_ADMIN_ROLE`, revert and redirect the user to `transferAdmin()`.
2. Consider removing the `renounceAdmin()` function to make sure that `_pendingAdmin` always finalizes to a successfully transferred, controlled admin address.

HYP-3 Potential Reverts Due to Underflows in `depositFromStaking()`

• Low ⓘ Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `d9e310f56a1c18aa5e1f1cbb8dd405b8c8c65ab5`.
The client provided the following explanation:

```
In WithdrawQueue::depositFromStaking we're now checking for potential underflows to the baseAmt and
rewardAmt parameters.
```

File(s) affected: `contracts/contract/WithdrawQueue.sol`

Description: In the `WithdrawQueue.depositFromStaking()` function is designed to on-demand top-up the vault if it does not hold sufficient funds to enable the fulfilling of the currently processing, pending request. The check in L364 `withdrawQueueAvailableAssets + ggAVAXAvailableAssets < req.expectedAssets` assures that the `WithdrawQueue` holds enough funds to do the top up.

`withdrawQueueAvailableAssets` includes the `msg.value` of this function invocation, so `baseAmt + rewardAmt` from the parameters, but the contract is also designed to hold funds outside of `msg.value`.

However, in the current design, the parameters `baseAmt` and `rewardAmt` are decreased based on the top-up that has been performed. This can lead to underflows, if e.g. the top up is covered not by `msg.value`, but by funds held by the `WithdrawQueue` contract outside of that. So while the initial check in L364 passes due to the the contract's balance outside of `msg.value`, the decrements of `baseAmt` and `rewardAmt` in L375 and L378, respectively, solely assume the funds were provided by `msg.value`.

The impact is that there could be frequent unintended reverts of the `depositForStaking()` flow, requiring manual investigation and adjustments of the parameters (and therefore the funds being deposited) to account for this bug.

Recommendation: Assuming that the queue's balance is also supposed to cover the top-ups, an underflow of the two parameters should be checked for and those values should only be set to zero if that is the case.

HYP-4 Consider Adding Rate Limiting Mechanisms to Protect Against Rogue Trusted Entities

• Low ⓘ Acknowledged

Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
We've decided not to address this concern at the moment, we will have sufficient protection for the
accounts that have access to call TokenggAVAX::withdrawForStaking. There are two methods, one that is
protected by onlyRegisteredNetworkContract("MinipoolManager") and a second that's protected by the
STAKER_ROLE. We trust our storage contract setup to protect an address from becoming a registered
network contract. And the STAKER_ROLE account will be carefully controlled, with external limits and
access features that will allow us to quickly disable if it were to be compromised.
```

File(s) affected: `contracts/contract/tokens/TokenggAVAX.sol`, `contracts/contract/WithdrawQueue.sol`

Description: The system design grants significant authority to the `MiniPoolManager` entities, including the theoretical ability for each to drain the entirety of the vault's underlying AVAX. While this risk is partially mitigated by multisig protections on sensitive operations, the core trust assumption remains strong: the system depends heavily on the integrity and uncompromised status of these managers.

In the event of key compromise, an attacker could submit malicious withdrawal requests and drain user funds without further protocol-level rate limiting or automated safeguards. The lack of enforced withdrawal limits, delay mechanisms to pause the protocol in a timely manner, or on-chain accounting for pool manager's withdrawals introduces a critical reliance on external trust.

Recommendation: To mitigate potential damage from compromised `MiniPoolManager` keys or rogue operator behaviour, we recommend implementing additional withdrawal controls at the protocol level: We recommend adding the `whenTokenNotPaused` modifier to the two `withdrawForStaking()` functions in the `TokenggAVAX` contract, limiting the damage a compromised privileged entity could have on the vault. Furthermore, we recommend a mandatory delay period (e.g., 10 hour) between a withdrawal for staking-request and its execution to enable monitoring and potential intervention. This would give the system admins sufficient time to review withdrawal requests and make use of the existing pause mechanic in case of suspicious activity. Furthermore, an per-entity withdrawal limit (daily/hourly/total) could be useful to cap the amount that any single manager can withdraw within a time window.

HYP-5 Risk of Duplicate Initialization

• Low ⓘ Acknowledged

Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
Our initializers don't perform any chained actions, so calling an unchained variant is not necessary.
```

File(s) affected: `contracts/contract/tokens/TokenggAVAX.sol`

Description: Directly invoking initializer functions instead of their unchained counterparts should be avoided to reduce the risk of duplicate initializations.

Recommendation: Use the unchained initializer functions in the `initialize()` and `reinitialize()` functions.

HYP-6 Minor Potential DoS vector on `requestsByOwner`

• Low ⓘ Fixed

Update
Marked as "Fixed" by the client.
Addressed in: `f3eff29180f0774f038611679a15bb6fd3d83589`.
The client provided the following explanation:

```
We implemented two fixes for this. First we followed the recommended remediation and added a minimum
share amount to requestUnstakeOnBehalfOf to prevent griefing. Second we added pagination to
getRequestsByOwner.
```

File(s) affected: `contracts/contract/WithdrawQueue.sol`

Description: Using the `requestUnstakeOnBehalfOf()`, users can make unstake requests on a `requester`'s behalf by covering the share token amount themselves. This then adds an entry to the `requester`'s individual `EnumerableSet` in the `requestsByOwner` mapping. While elements can be added and removed in O(1) time, enumerating the array is done in linear time. The code only leverages such an enumeration in the `cancelRequests()` function, which enables a user to cancel their own set of requests by iterating through the `requestsByOwner` values instead of calling `cancelRequest()` multiple times. However, external contracts might interact with this list more heavily, as it is also exposed as an `external view` function using `getRequestsByOwner()`, which could theoretically a more severe DoS vector on contracts outside of the scope of this audit.

Recommendation: Consider adding a reasonable minimum size to the `shares` parameter in the `requestUnstakeOnBehalfOf()` to make such spam attacks economically unfeasible.

HYP-7
Assets Intended for ggAvax Holders Repurposed for Fulfilling Withdrawal Requests in Certain Cases

• Low ⓘ Fixed

Update
Marked as "Fixed" by the client.
Addressed in: `2066c802fcbd2e573c28e402f81e62804115bafd` and `8b285ab56713dbbe8f534aaa4012ff76d42cab99`.
The client provided the following explanation:

```
Followed the suggested remediation and are always donating excessAVAX
```

File(s) affected: `contracts/contract/WithdrawQueue.sol`

Description: Within the `depositFromStaking()` function, the function intended to allocate funds to pending withdrawal requests, the local variable `excessAVAX` is calculated as a form of sum of penalty yield of all the currently processed withdrawal requests intended to benefit `ggAVAX` holders by being donated back into the vault. However, if the function returns early (e.g., due to hitting a limit condition such as at line 414 or the graceful return in L365), any accumulated `excessAVAX` is not explicitly donated back to the protocol or users. Because `excessAVAX` is a local variable, its value is discarded upon return, and there is no mechanism to later on donate those funds.

Although the `excessAVAX` is not becoming stuck in the contract, it is repurposed with no longer any benefit to the `ggAVAX` holders, such as potentially being redirected to fulfil unrelated withdrawals, effectively repurposing `ggAVAx` holder's yield in favor of the staking providers.

Recommendation: Ensure that any `excessAVAX` accrued is always properly donated or otherwise handled in a way that aligns with its intended purpose as yield for `ggAVAX` holders.

HYP-8
Inconsistent Application of Sanity Checks on `withdrawForStaking(s)` Functions

• Low ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
We are checking the amountAvailableForStaking in the MinipoolManager where it calls to
TokenggAVAX::withdrawForStaking.
```

File(s) affected: `contracts/contract/tokens/TokenggAVAX.sol`

Description: The `withdrawForStaking()` function intended for the `STAKER_ROLE` contains a check that ensures certain amount of `AVAX` is not withdrawable for yield generating purposes. This is done by checking that assets to be withdrawn is not greater than `ggAVAX.amountAvailableForStaking()`. However, this check is missing in the separate `TokenggAVAX.withdrawForStaking()` function intended for the `MinipoolManager` role.

Recommendation: Have that check present in both `withdrawForStaking()` functions.

HYP-9 `depositFromStaking()` Can Revert in Certain States

• Low ⓘ Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `81c5483c733b50467b39b43879782f4fede051f1`.
The client provided the following explanation:

```
We followed the remediation, and are now calculating and covering the fee that TokenggAVAX takes when
depositing rewards
```

File(s) affected: `contracts/contract/WithdrawQueue.sol`

Description: The `WithdrawQueue.depositFromStaking()` function finalizes pending withdrawal requests by essentially allocating funds to cover the withdrawal request. It includes a just-in-time liquidity top-up of the vault's underlying asset that essentially makes the vault solvent enough to cover the `redeemAVAX()` call the queue is about to perform. This missing liquidity is covered by the proportional contribution from base and reward amount from the repayment.

However, if the `protocolDAO` happens to have set a non-zero fee, a cut is taken from the `rewardAmt` forwarded as part of the top-up in `tokenggAVAX.depositFromStaking()`. This causes less than `amountToDeposit` to actually be deposited, resulting the `redeemAVAX()` call in L387 to revert.

Recommendation: Either make `amountToDeposit` be solely covered by the returned base asset component, or apply the fee to the `rewardAmt` when calling `_depositToGGAVAX()`.

HYP-10 Potential Reverts of the `getExcessShares()` function due to Underflow

• Low ⓘ Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `65bb1dfcb2f37faa6d4af0b9e27a000a02a68709`.
The client provided the following explanation:

```
We followed the recommended remediation and fixed by returning 0 when pstAVAXVaultShares *
ggAVAXTotalAssets < totalPstTokens * ggAVAXTotalShares
```

File(s) affected: `contracts/contract/tokens/TokenpstAVAX.sol`

Description: Our tests of some edge cases have shown that, over time, the roundings in favour of the protocol of the share token exchange rate materialize to the equation defined in the `getExcessShares()` function to possibly revert due to an underflow, as `pstAVAXVaultShares * ggAVAXTotalAssets` can be smaller than `totalPstTokens * ggAVAXTotalShares`. While we have only gotten the difference to be very minor, it was enough to cause functions like the `stripYield()` function to revert, which would have only been fixable with donations to the `ggAVAX` vault.

Recommendation: In the `getExcessShares()` function, return `0` when `pstAVAXVaultShares * ggAVAXTotalAssets < totalPstTokens * ggAVAXTotalShares`.

HYP-11 Improvements to `try-catch` Logic in `depositFromStaking`

• Informational ⓘ Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `0307bf0c76b239ea7e8bdea64a15dc7b03c7ec79`.
The client provided the following explanation:

```
Followed the recommended remediation and throw a new error from TokenggAVAX::redeemAVAX that we check
for in WithdrawQueue::depositFromStaking. That error is now the only graceful way to exit the loop
```

File(s) affected: `contracts/contract/WithdrawQueue.sol`, `contracts/contract/tokens/TokenggAVAX.sol`

Description: Function `WithdrawQueue.depositFromStaking()` is responsible for depositing AVAX in the contract that can be used to fulfill the pending unstake requests, at L#387 we have a `try-catch` statement that calls `tokenggAVAX.redeemAVAX` which is trying to redeem all shares from a request, `catch` part of the stamenet doesn't have any filters and the assumption is that this call can fail only when there is not enough liquidity to fulfill a request. However that is not completely true as there can be a gas edge-case thanks to the 63/64 gas rule so in some very isolated and rare scenario, it can fail for other reasons even if there is enough liqudity.

Recommendation: In `tokenggAVAX.redeemAVAX()`. check that `IWAVAX(address(asset)).balanceOf(address(this)) > assets` and revert with a custom error before the `IWAVAX(address(asset)).withdraw(assets);` call. That custom revert should be the only graceful exit of the for-loop.

Auditor Suggestions

S1 Changes to the Pause-Modifier

Acknowledged

Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
We're not changing the pause interaction here
```

File(s) affected: `contracts/contract/tokens/TokenggAVAX.sol`

Description: The `whenTokenNotPaused()` modifier passes, even if the contract is paused, for `amt = 0`. This can lead to possibly confusing event and emission non-blocked contract interactions in a paused contract state.

Recommendation: Consider removing the `amt > 0` check from the equation.

S2 Assure `maxPendingRequestsLimit` Is Used Properly

Fixed

Update

Marked as "Fixed" by the client.
Addressed in: `ac67e792a60b3b7ca6badc6d623331202e82bd80`.
The client provided the following explanation:

```
This variable was poorly named, it actually refers to the amount of requests we process in one deposit
in WithdrawQueue::depositFromStaking. The variable has been renamed to maxRequestsPerStakingDeposit
```

File(s) affected: `contracts/contract/WithdrawQueue.sol`

Description: The `maxPendingRequestsLimit` storage variable either has a misleading name or is being used incorrectly. Currently, its value is used in the `depositFromStaking()` function to cap the number of requests being processed during each invocation of the function, a vital protection against block gas limit-concerns. However, please note that this won't cap the maximum possible number of processed requests overall, but just the number of requests that can be processed in each invocation of the function.

The variable name, however, suggests that it should be used in `requestUnstake()` and `requestUnstakeOnBehalfOf()` to cap the total number of pending withdrawal requests at any given time.

Recommendation: Reevaluate the purpose of this variable. If it is intended to control batching during processing (as in `depositFromStaking()`), then rename it accordingly. If it is meant to cap the number of pending requests, update the logic in `requestUnstake()` and `requestUnstakeOnBehalfOf()` to enforce that limit. Also, ensure it is initialized with a proper value in the contract's initializer function.

S3 Remove Duplicate Code by Using Internal Functions

Fixed

Update
Marked as "Fixed" by the client.
Addressed in: `8325601d8cd45e54d806abfd42d337635c139eea`.
The client provided the following explanation:

```
Following the recommended remediation we've created internal functions to combine the logic in
TokenggAVAX::depositFromStaking, TokenggAVAX::depositFromYield and additionally
WithdrawQueue::requestUnstake and WithdrawQueue::requestUnstakeOnBehalfOf
```

File(s) affected: `contracts/contract/tokens/TokenggAVAX.sol`, `contracts/contract/WithdrawQueue.sol`

Description: The two different overloads of `depositFromStaking()` as well as the `depositYield()` function have very similar implementations. Considering the contracts are upgradeable, this kind of code duplication can become troublesome in the future—especially if changes in business logic require updates to these implementations. Any such change would need to be reflected in all similar functions, increasing the risk of inconsistency and bugs.

Additionally, the functions `requestUnstakeOnBehalfOf()` and `requestUnstake()` also share very similar logic.

Recommendation: Consider introducing new internal functions that encapsulate the shared logic of these functions. Each external function can then call the internal function with the appropriate parameters. This will improve maintainability, reduce redundancy, and minimize the risk of errors when changes are made.

S4 Uncallable Functions Defined

Acknowledged

Update
Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
We're leaving these functions defined in TokenggAVAX for future implementations of WithdrawQueue that
might utilize them.
```

File(s) affected: `contracts/contract/tokens/TokenggAVAX.sol`

Description: The `TokenggAVAX` contract defines a few functions that are not callable: `withdrawAVAX()`, `withdraw()`, `redeem()` are only callable by the role solely intended to be held by the `WithdrawQueue` contract, yet it is an external call the `WithdrawQueue` contract has no capability to perform. As the `WithdrawQueue` is an upgradeable contract with the theoretical risk of private key compromise, this imposes, in our opinion, an unnecessary risk.

Recommendation: Consider removing the `withdrawAVAX()` function and overriding the inherited `withdraw()` and `redeem()` function to solely revert.

S5 General Code Improvements

Fixed

Update
Marked as "Fixed" by the client.
Addressed in: `d99453e7d6191ce2dd6cff67a1832e7393466fed`.
The client provided the following explanation:

```
Considered and resolved all four bullet points outlined.
  Unnecessary cleanup of data to be deleted in WithdrawQueue::cancelRequest and
WithdrawQueue::claimUnstake
  Removed assignment of req.requester in WithdrawQueue::cancelRequest
  TokenggAVAX::withdrawForStaking fixed the external self call in fetching amountAvailableForStaking
  Changed various function calls that aren't used internally from public to external
```

File(s) affected: `contracts/contract/WithdrawQueue.sol`, `contracts/contract/tokens/TokenggAVAX.sol`, `contracts/contract/tokens/TokenpstAVAX.sol`

Description: A few suggestions for general code improvements have been identified:

1. There's no need to manually reset properties of `UnstakeRequest` such as `req.allocatedFunds` in functions like `WithdrawQueue.cancelRequest()` or `WithdrawQueue.claimUnstake()`. Since the `request` is deleted from storage from the `requests` mapping in all these cases, the resets are redundant.
2. In `WithdrawQueue.cancelRequest()`, the value of `req.requester` is saved to a local variable under the assumption that it might be lost when the request is removed. However, in this function, `req.requester` is equal to `msg.sender`, so the local copy is unnecessary

and can be safely removed.
3. In `TokenGGAvax.withdrawForStaking()`, effectively an external self-call is performed by fetching the contract's own address and calling `ggAVAX.amountAvailableForStaking()`. This is unnecessary, as the `amountAvailableForStaking()` function is already marked as `public`, so can be called internally.
4. Functions not called within a contract, but marked as `public` can be marked as `external` to save some gas, e.g. regarding the role management functions in the `TokenggAVAX` contract.

Recommendation: Consider applying the above recommendations to improve clarity, safety, and maintainability of the code.

S6 Improvements to Event Emission/Error Handling

Acknowledged

Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
We've elected not to make any changes to the events suggested here.
  TokenggAVAX::amountAvailableForStaking over-allocation is implicitly understood if the
amountAvailbleForStaking returns as 0
  TokenggAVAX::withdrawAVAX, redeemAVAX I'd rather not pass the requester through to these functions
for the particular WithdrawQueue use case
  TokenpstAVAX::getExcessShares problematic liquidity scenario where ggAVAXTotalAssets <=
totalPstTokens can happen when there have been no rewards to TokenggAVAX, and returning 0 excess shares
in that case is appropriate
  WithdrawQueue::claimUnstake event for totalAllocatedFunds -= amount underflowing should never happen
as an invariant of the protocol, and if it does a revert due to underflow is appropriate to investigate
the issue further.
```

File(s) affected: `contracts/contract/tokens/TokenggAVAX.sol`, `contracts/contract/WithdrawQueue.sol`, `contracts/contract/tokens/TokenpstAVAX.sol`

Description: A few improvements can be made to overall event emission and error handling:

- In `TokenggAVAX.amountAvailableForStaking()`, consider emitting an event for transparency and alert of over-allocation in case of `reservedAssets + stakingTotalAssets > totalAssets_)`.
- The event emission in `withdrawAVAX()` and `redeemAVAX()` could be improved if the `requester` performing the queue interaction leading to the withdrawal were to be forwarded to those functions and emitted for the `caller` and `receiver` fields of the `Withdraw` and `Redeem` events.
- In `TokenpstAVAX.getExcessShares()`, an event should be emitted in case of `ggAVAXTotalAssets <= totalPstTokens` as it indicates problematic liquidity.
- In `WithdrawQueue.claimUnstake()`, a custom `VaultCurrentlyNotSufficientlySolvent` revert should be introduced in case `totalAllocatedFunds -= amount` causes an underflow.

Recommendation: Consider implementing these suggestions.

Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – 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 pose an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.
- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **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).

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.

Files

Repo: `https://github.com/multisig-labs/gogopool`

- `889...38f ./.editorconfig`
- `4dd...9c7 ./.env.example`
- `1c8...9ef ./.eslintignore`
- `76f...7bb ./.eslintrc.js`
- `dd0...814 ./.github/workflows/add_issue_to_pm.yml`
- `bf8...8ae ./.github/workflows/unit_test.yaml`
- `ced...f4f ./.gitignore`
- `bb9...95b ./.gitmodules`
- `729...652 ./.npmignore`
- `d0f...77b ./.prettierignore`
- `053...ea3 ./.prettierrc.json`
- `c50...72b ./.solhint.json`
- `16d...c97 ./.solhintignore`
- `624...de3 ./.vscode/README.md`
- `f1d...961 ./.vscode/extensions.example.json`
- `0a3...a78 ./.vscode/launch.example.json`
- `da0...a73 ./.vscode/settings.example.json`
- `167...7e2 ./Justfile`
- `848...7ef ./LICENSE`
- `2a8...316 ./README.md`
- `0a9...a82 ./contracts/contract/Base.sol`
- `f4d...e33 ./contracts/contract/BaseAbstract.sol`
- `e2e...a5b ./contracts/contract/BaseUpgradeable.sol`
- `20c...9f5 ./contracts/contract/ClaimNodeOp.sol`
- `bb7...d77 ./contracts/contract/ClaimProtocolDAO.sol`
- `58a...9d8 ./contracts/contract/MinipoolManager.sol`
- `14f...72d ./contracts/contract/MinipoolStreamliner.sol`
- `c8e...2b9 ./contracts/contract/MultisigManager.sol`
- `964...a99 ./contracts/contract/Ocyticus.sol`
- `d58...6bf ./contracts/contract/Oracle.sol`
- `24e...f83 ./contracts/contract/ProtocolDAO.sol`
- `299...66b ./contracts/contract/RewardsPool.sol`
- `d17...5af ./contracts/contract/Staking.sol`
- `58d...3bf ./contracts/contract/Storage.sol`
- `b1c...7ba ./contracts/contract/Timelock.sol`
- `bd0...3f3 ./contracts/contract/TwapGGP.sol`
- `7c7...bd9 ./contracts/contract/Vault.sol`
- `619...ddc ./contracts/contract/WithdrawQueue.sol`
- `145...90d ./contracts/contract/hardwareProviders/AvalancheHardwareRental.sol`
- `ba3...88d ./contracts/contract/hardwareProviders/CoqnetHardwareRental.sol`
- `680...f07 ./contracts/contract/hardwareProviders/SubnetHardwareRentalBase.sol`
- `5f8...84d ./contracts/contract/hardwareProviders/SubnetHardwareRentalMapping.sol`
- `37c...8e9 ./contracts/contract/previousVersions/IHardwareProvider.sol`
- `6c1...496 ./contracts/contract/previousVersions/MinipoolStreamlinerV1.sol`
- `d29...4c4 ./contracts/contract/previousVersions/MinipoolStreamlinerV2.sol`
- `935...e01 ./contracts/contract/previousVersions/TokenggAVAXV1.sol`
- `7e2...2db ./contracts/contract/previousVersions/TokenggAVAXV2.sol`
- `1b1...92d ./contracts/contract/tokens/GGAVAXRateProvider.sol`

- `7b7...644 ./contracts/contract/tokens/TokenGGP.sol`
- `1df...866 ./contracts/contract/tokens/TokenggAVAX.sol`
- `a0a...837 ./contracts/contract/tokens/TokenpstAVAX.sol`
- `900...af4 ./contracts/contract/tokens/upgradeable/ERC20Upgradeable.sol`
- `9e0...c40 ./contracts/contract/tokens/upgradeable/ERC4626Upgradeable.sol`
- `571...739 ./contracts/contract/utils/CREATE3Factory.sol`
- `abf...bdf ./contracts/contract/utils/Multicall.sol`
- `6ca...565 ./contracts/contract/utils/Multicall3.sol`
- `4b2...6b3 ./contracts/contract/utils/OneInchMock.sol`
- `943...4e3 ./contracts/contract/utils/RialtoSimulator.sol`
- `e59...ff8 ./contracts/contract/utils/WAVAX.sol`
- `887...631 ./contracts/interface/IERC20.sol`
- `718...20c ./contracts/interface/ILBRouter.sol`
- `51d...0af ./contracts/interface/IOneInch.sol`
- `0fc...79e ./contracts/interface/IOonodzEntryPoint.sol`
- `977...d91 ./contracts/interface/IUniswapV2Pair.sol`
- `e91...d14 ./contracts/interface/IWAVAX.sol`
- `b24...4c8 ./contracts/interface/IWithdrawer.sol`
- `f59...4dc ./contracts/types/MinipoolStatus.sol`
- `ad2...518 ./cspell-ignore-words.txt`
- `541...c1d ./cspell.json`
- `f69...f8b ./deployed/1337-addresses.tmpl.json`
- `c55...fa4 ./deployed/31337-addresses.tmpl.json`
- `ac2...ff3 ./deployed/43112-addresses.tmpl.json`
- `2fb...b03 ./deployed/43113-addresses.json`
- `e22...299 ./deployed/43113-addresses.tmpl.json`
- `c31...8cd ./deployed/43114-addresses.json`
- `93b...576 ./deployed/43114-addresses.tmpl.json`
- `589...f17 ./foundry.toml`
- `ed9...5c8 ./generated/addresses/43113.ts`
- `3af...e2a ./generated/addresses/43114.ts`
- `d8f...ca0 ./generated/contracts/ArtifactHardwareProvider.ts`
- `93e...3a3 ./generated/contracts/ClaimNodeOp.ts`
- `dbf...def ./generated/contracts/MinipoolManager.ts`
- `44c...4d2 ./generated/contracts/MinipoolStreamliner.ts`
- `f65...699 ./generated/contracts/OneInchMock.ts`
- `79c...403 ./generated/contracts/Oracle.ts`
- `f34...09b ./generated/contracts/ProtocolDAO.ts`
- `f75...aa6 ./generated/contracts/RewardsPool.ts`
- `1e9...616 ./generated/contracts/Staking.ts`
- `63a...fc0 ./generated/contracts/Storage.ts`
- `7db...5ee ./generated/contracts/TokenGGP.ts`
- `b8f...6f9 ./generated/contracts/TokenggAVAX.ts`
- `780...c0d ./generated/errors.ts`
- `9e7...a5b ./hardhat.config.ts`
- `464...632 ./package.json`
- `37d...e79 ./remappings.txt`
- `fef...42e ./scenarios/Full Cycle Multi Users.md`
- `ffe...905 ./scenarios/Full Cycle One User.md`
- `40f...a6a ./scenarios/Overview.md`
- `004...6e2 ./script/EnvironmentConfig.s.sol`
- `a0c...393 ./script/README.md`
- `c4c...977 ./script/change-minipool.s.sol`
- `d8f...0fd ./script/coqnet/add-coqnet-mainnet-hardware-providers.s.sol`
- `f58...176 ./script/coqnet/deploy-avalanche-hardware-provider.s.sol`

- `91b...298 ./script/coqnet/deploy-coqnet-hardware-provider.s.sol`
- `41c...452 ./script/coqnet/deploy-hardware-rental-mapping.s.sol`
- `7c8...7ba ./script/coqnet/remove_deployer_role.s.sol`
- `b87...e3e ./script/coqnet/set-coqnet-avax-price-feed.s.sol`
- `1a4...93d ./script/coqnet/set_mainnet_configs.s.sol`
- `8f1...bfc ./script/deploy-tokenggavax.s.sol`
- `216...966 ./script/get-ggp-rewards.s.sol`
- `639...977 ./script/init-dev.s.sol`
- `c27...ff0 ./script/init-fuji.s.sol`
- `012...a56 ./script/init-mainnet.s.sol`
- `0ef...d6c ./script/init-rialto.s.sol`
- `e75...a2a ./script/obsolete/deploy-2023-09-05.s.sol.old`
- `bb8...d18 ./script/obsolete/deploy-artifact-hardware.s.sol.old`
- `f6f...87f ./script/obsolete/deploy-contract.s.sol.old`
- `dda...151 ./script/obsolete/deploy-minipool-streamlinerv2.s.sol.old`
- `41a...9a3 ./script/obsolete/deploy-mp-redesign.s.sol.old`
- `744...ba6 ./script/obsolete/deploy-rate-provider.s.sol.old`
- `5f0...e1f ./script/obsolete/deploy-timelock.s.sol.old`
- `fa4...610 ./script/obsolete/deploy.s.sol.old`
- `c93...20c ./script/obsolete/deploy_relaunch_minipool.s.sol.old`
- `ae5...0cd ./script/obsolete/doctor.s.sol.old`
- `23e...8b3 ./script/obsolete/hardware-provider-upgrade.s.sol.old`
- `26d...338 ./script/rollback_fuji.s.sol`
- `2fe...d29 ./script/run-rialto.s.sol`
- `3cd...c5a ./script/set-guardian.s.sol`
- `7ef...e84 ./script/test-upgrade-specified-impl.sol`
- `6c1...c80 ./script/upgrade-liquid-staking-system.s.sol`
- `8cd...06e ./script/upgrade-minipool-streamliner.s.sol`
- `518...132 ./script/upgrade-tokenggavax.s.sol`
- `0e7...82c ./slither.config.json`
- `084...db2 ./tasks/README.md`
- `51c...eed ./tasks/dao.js`
- `c64...1d7 ./tasks/debug.js`
- `927...ba6 ./tasks/inflation.js`
- `246...f47 ./tasks/lib/utils.js`
- `a53...c30 ./tasks/minipool.js`
- `809...bd2 ./tasks/multisig.js`
- `cc8...71a ./tasks/nopClaim.js`
- `9de...369 ./tasks/oracle.js`
- `a62...b36 ./tasks/staking.js`
- `7c2...443 ./tasks/tokens.js`
- `61b...094 ./tasks/vault.js`
- `2ad...5d2 ./test/integration/ChainHardwareRentalBaseIntegration.t.sol`
- `208...7a3 ./test/integration/TestFailure.sol`
- `ab3...946 ./test/invariant/WithdrawQueueHandler.sol`
- `9ec...25d ./test/invariant/WithdrawQueueInvariants.t.sol`
- `1c6...0a8 ./test/unit/AVAXHighWaterTest.t.sol`
- `309...3ce ./test/unit/BaseContractTest.sol`
- `aba...e63 ./test/unit/ClaimNodeOp.t.sol`
- `75a...c7f ./test/unit/ClaimProtocolDAO.t.sol`
- `4fa...e20 ./test/unit/Delegation.t.sol`
- `c4e...ca2 ./test/unit/ERC20Upgradeable.t.sol`
- `f99...541 ./test/unit/ERC4626Std.t.sol`
- `cee...db7 ./test/unit/ERC4626X.t.sol`
- `6a4...819 ./test/unit/MEVRewards.t.sol`

- `a85...afa ./test/unit/MinipoolManager.t.sol`
- `32f...0b4 ./test/unit/MinipoolStreamliner.t.sol`
- `ca3...34c ./test/unit/MinipoolStreamlinerV2.t.sol`
- `27a...28a ./test/unit/MultisigManager.t.sol`
- `f97...b21 ./test/unit/Ocyticus.t.sol`
- `ec8...c22 ./test/unit/OneInchMock.t.sol`
- `1e2...fc6 ./test/unit/Oracle.t.sol`
- `425...3dc ./test/unit/ProtocolDAO.t.sol`
- `231...436 ./test/unit/README.md`
- `a07...ddf ./test/unit/RateProvider.t.sol`
- `b25...dee ./test/unit/RewardsPool.t.sol`
- `337...85b ./test/unit/Scenarios.t.sol`
- `0ae...7b3 ./test/unit/Staking.t.sol`
- `8e0...ce6 ./test/unit/Storage.t.sol`
- `74f...211 ./test/unit/SubnetHardwareRentalBaseTest.t.sol`
- `057...abc ./test/unit/Timelock.t.sol`
- `fdf...5d9 ./test/unit/TokenGGP.t.sol`
- `120...447 ./test/unit/TokenggAVAX.t.sol`
- `52f...37b ./test/unit/TokenggAVAXAccessControl.t.sol`
- `f18...1ef ./test/unit/TokenggAVAXFirstDepositor.t.sol`
- `4da...fb5 ./test/unit/TokenpstAVAXTest.t.sol`
- `2d1...db9 ./test/unit/TwapGGP.t.sol`
- `26a...225 ./test/unit/Vault.t.sol`
- `864...d7d ./test/unit/WithdrawQueue.t.sol`
- `9d0...6fd ./test/unit/WithdrawQueueGasTests.t.sol`
- `6db...08b ./test/unit/upgrade/MinipoolStreamlinerUpgrade.t.sol`
- `8cf...118 ./test/unit/upgrade/TokenUpgradeTests.t.sol`
- `485...a34 ./test/unit/utils/BaseTest.sol`
- `680...9fd ./test/unit/utils/ERC20UpgradeableDangerous.sol`
- `be0...5a8 ./test/unit/utils/ERC20UpgradeableSafe.sol`
- `28c...ec9 ./test/unit/utils/ERC4626UpgradeableDangerous.sol`
- `12d...8d2 ./test/unit/utils/ERC4626UpgradeableSafe.sol`
- `743...5ab ./test/unit/utils/MockChainlink.sol`
- `7c3...6d3 ./test/unit/utils/MockERC20Upgradeable.sol`
- `01a...65b ./test/unit/utils/MockHardwareProvider.sol`
- `32d...306 ./test/unit/utils/MockSubnetHardwareRental.sol`
- `263...813 ./test/unit/utils/MockTokenggAVAXV2.sol`
- `e12...5c9 ./test/unit/utils/MockTokenggAVAXV2Dangerous.sol`
- `b4d...181 ./test/unit/utils/MockTokenggAVAXV2Safe.sol`
- `956...ca4 ./test/unit/utils/MockTraderJoeRouter.sol`
- `d40...b43 ./tsconfig.json`
- `7db...9f4 ./yarn.lock`

Test Suite Results

Tests were executed with `forge coverage`. The failing tests can solely be attributed to us not having set up the local fork setup.

Fix-Review Update Additional tests have been added to validate some of the findings.

```
Ran 1 test for test/unit/ClaimProtocolDAO.t.sol:ClaimProtocolDAOTest
[PASS] testSpendFunds() (gas: 149216)
Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 100.58ms (36.29ms CPU time)

Ran 33 tests for test/unit/ProtocolDAO.t.sol:ProtocolDAOTest
```

```
[PASS] testGetClaimingContractPct() (gas: 23732)
[PASS] testGetExpectedAVAXRewardsRate() (gas: 13167)
[PASS] testGetInflation() (gas: 16918)
[PASS] testGetInflationIntervalRate() (gas: 15056)
[PASS] testGetInflationIntervalSeconds() (gas: 13080)
[PASS] testGetMaxCollateralizationRatio() (gas: 13125)
[PASS] testGetMinCollateralizationRatio() (gas: 13145)
[PASS] testGetMinipoolCancelMoratoriumSeconds() (gas: 13213)
[PASS] testGetMinipoolMaxAVAXAssignment() (gas: 13166)
[PASS] testGetMinipoolMinAVAXAssignment() (gas: 13125)
[PASS] testGetMinipoolMinAVAXStakingAmt() (gas: 13122)
[PASS] testGetMinipoolNodeCommissionFeePct() (gas: 13079)
[PASS] testGetRewardsCycleSeconds() (gas: 13211)
[PASS] testGetRewardsEligibilityMinSeconds() (gas: 13102)
[PASS] testGetTargetGGAVAXReserveRate() (gas: 13146)
[PASS] testPauseContract() (gas: 103906)
[PASS] testRegisterContract() (gas: 157478)
[PASS] testRegisterContractAlreadyRegistered() (gas: 124146)
[PASS] testRegisterContractInvalid() (gas: 59083)
[PASS] testRegisterContractNotGuardian() (gas: 40591)
[PASS] testResumeContract() (gas: 40504)
[PASS] testSetClaimingContractPct() (gas: 70342)
[PASS] testSetClaimingContractPctGreaterThanOne() (gas: 27158)
[PASS] testSetExpectedAVAXRewardsRate() (gas: 48413)
[PASS] testSetExpectedAVAXRewardsRateGreaterThanOne() (gas: 31766)
[PASS] testSetFeeBips() (gas: 59323)
[PASS] testUnregisterContract() (gas: 131201)
[PASS] testUnregisterContractNotGuardian() (gas: 17139)
[PASS] testUpgradeContract() (gas: 186428)
[PASS] testUpgradeContractExistingNotRegistered() (gas: 137134)
[PASS] testUpgradeContractInvalid() (gas: 125630)
[PASS] testUpgradeContractNotGuardian() (gas: 41599)
[PASS] testUpgradeProtocolDAO() (gas: 2201023)
Suite result: ok. 33 passed; 0 failed; 0 skipped; finished in 149.78ms (77.49ms CPU time)

Ran 4 tests for test/unit/AVAXHighWaterTest.t.sol:AVAXStateVariableTest
[PASS] testDifferentAVAXAssignment() (gas: 2767762)
[PASS] testMultipleMinipools() (gas: 3167319)
[PASS] testQueuedMinipools() (gas: 2686707)
[PASS] testRewardsReset() (gas: 3485821)
Suite result: ok. 4 passed; 0 failed; 0 skipped; finished in 426.64ms (355.03ms CPU time)

Ran 18 tests for test/unit/BaseContractTest.sol:BaseContractTest
[PASS] testAddUint() (gas: 33161)
[PASS] testAddress() (gas: 23316)
[PASS] testBool() (gas: 23174)
[PASS] testBytes() (gas: 27130)
[PASS] testBytes32() (gas: 22882)
[PASS] testGetContractAddress() (gas: 32418)
[PASS] testGetContractName() (gas: 34782)
[PASS] testGuardianOrRegisteredContractContract() (gas: 66044)
[PASS] testGuardianOrSpecificRegisteredContractContract() (gas: 37733)
[PASS] testInt() (gas: 22825)
[PASS] testOnlyGuardian() (gas: 51941)
[PASS] testOnlyMultisig() (gas: 167186)
[PASS] testOnlyRegisteredNetworkContract() (gas: 52447)
[PASS] testOnlySpecificRegisteredContract() (gas: 31383)
[PASS] testString() (gas: 26900)
[PASS] testSubUint() (gas: 33203)
[PASS] testUint() (gas: 22923)
[PASS] testWhenNotPaused() (gas: 63728)
Suite result: ok. 18 passed; 0 failed; 0 skipped; finished in 65.76ms (14.50ms CPU time)

Ran 1 test for test/unit/RateProvider.t.sol:RateProviderTest
[PASS] testRate() (gas: 27793)
Logs:
  1133113058981626385

Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 2.02s (1.04s CPU time)

Ran 10 tests for test/unit/RewardsPool.t.sol:RewardsPoolTest
[PASS] testGetClaimingContractDistribution() (gas: 455645)
```

```
[PASS] testGetInflationIntervalsElapsed() (gas: 37105)
[PASS] testInflationAmtWithRewardsDelay() (gas: 448426)
[PASS] testInflationCalculate() (gas: 218404)
[PASS] testInitialization() (gas: 22003)
[PASS] testMaxInflation() (gas: 9095460)
[PASS] testMultipleMultisigRewards() (gas: 838744)
Logs:
  testGasCreateMinipool Gas: 529658

[PASS] testStartRewardsCycle() (gas: 501182)
[PASS] testStartRewardsCyclePaused() (gas: 90160)
[PASS] testZeroMultisigRewards() (gas: 484366)
Suite result: ok. 10 passed; 0 failed; 0 skipped; finished in 617.02ms (578.52ms CPU time)

Ran 11 tests for test/unit/Scenarios.t.sol:ScenariosTest
[PASS] testAVAXValidatingHighWaterMarkCancelledMinipool() (gas: 2464456)
[PASS] testChangeInflationRate() (gas: 888361)
[PASS] testFullCycleHappyPath() (gas: 2469918)
[PASS] testFullCycleNoRewards() (gas: 2501068)
[PASS] testGGPRewardsForFinishedMinipoolsUnderCollat() (gas: 2384907)
[PASS] testGGPRewardsForWithdrawableMinipoolsUnderCollat() (gas: 2354260)
[PASS] testHalfRewardsForUnvestedGGPLargerScale() (gas: 9911652)
[PASS] testHalfRewardsForUnvestedGGPSmallScale() (gas: 5318118)
[PASS] testRewardsManipulation() (gas: 4092671)
[PASS] testStakeMinipoolUnstakeStakeScenario() (gas: 2390381)
[PASS] testStakingGGPOnly() (gas: 632319)
Suite result: ok. 11 passed; 0 failed; 0 skipped; finished in 844.60ms (814.96ms CPU time)

Ran 33 tests for test/unit/Staking.t.sol:StakingTest
[PASS] testAVAXValidating() (gas: 233186)
[PASS] testAVAXValidatingHighWaterMark() (gas: 274327)
[PASS] testDecreaseAVAXAssigned() (gas: 963409)
[PASS] testDecreaseAVAXStake() (gas: 963483)
[PASS] testDecreaseGGPRewards() (gas: 253130)
[PASS] testGetAVAXAssigned() (gas: 956952)
[PASS] testGetAVAXStake() (gas: 1704847)
[PASS] testGetCollateralizationRatio() (gas: 968206)
[PASS] testGetEffectiveGGPStaked() (gas: 1590480)
[PASS] testGetEffectiveGGPStakedWithLowGGPPrice() (gas: 1610489)
[PASS] testGetGGPRewards() (gas: 2079771)
[PASS] testGetGGPStake() (gas: 333601)
[PASS] testGetLastRewardsCycleCompleted() (gas: 2087319)
[PASS] testGetMinimumGGPStake() (gas: 1530309)
[PASS] testGetRewardsStartTime() (gas: 960556)
[PASS] testGetStaker() (gas: 387438)
[PASS] testGetStakerCount() (gas: 331658)
[PASS] testGetTotalGGPStake() (gas: 379612)
[PASS] testIncreaseAVAXAssigned() (gas: 963389)
[PASS] testIncreaseAVAXStake() (gas: 963506)
[PASS] testIncreaseGGPRewards() (gas: 242938)
[PASS] testRestakeGGP() (gas: 316217)
[PASS] testSetRewardsStartTime() (gas: 246441)
[PASS] testSetRewardsStartTimeInvalid() (gas: 221642)
[PASS] testSlashGGP() (gas: 246312)
[PASS] testStakeAndWithdrawGGPPaused() (gas: 294559)
[PASS] testStakeGGP() (gas: 215287)
[PASS] testStakeOnBehalfOfGGPWithLock() (gas: 310335)
[PASS] testStakeOnBehalfOfLockRecentTimestamp() (gas: 284956)
[PASS] testStakeOnBehalfOfMustBeAuthorized() (gas: 246355)
[PASS] testStakeOnBehalfOfNoLock() (gas: 290216)
[PASS] testStakeWithdraw() (gas: 1081933)
[PASS] testWithdrawGGP() (gas: 243190)
Suite result: ok. 33 passed; 0 failed; 0 skipped; finished in 320.35ms (292.55ms CPU time)

Ran 7 tests for test/unit/ERC4626X.t.sol:xERC4626Test
[PASS] testSyncRewardsAfterEmptyCycle(uint128,uint128) (runs: 256, μ: 237593, ~: 237593)
[PASS] testSyncRewardsAfterFullCycle(uint128,uint128,uint128) (runs: 256, μ: 239061, ~: 239061)
[PASS] testSyncRewardsFailsDuringCycle(uint128,uint128,uint256) (runs: 256, μ: 205778, ~: 205851)
[PASS] testTotalAssetsAfterDeposit(uint128,uint128) (runs: 256, μ: 181171, ~: 181171)
[PASS] testTotalAssetsAfterWithdraw(uint128,uint128) (runs: 256, μ: 201965, ~: 201965)
[PASS] testTotalAssetsDuringDelayedRewardDistribution(uint128,uint128) (runs: 256, μ: 294624, ~: 294624)
[PASS] testTotalAssetsDuringRewardDistribution(uint128,uint128) (runs: 256, μ: 295456, ~: 295456)
```

```
Suite result: ok. 7 passed; 0 failed; 0 skipped; finished in 4.04s (3.97s CPU time)

Ran 3 tests for test/unit/Storage.t.sol:StorageTest
[PASS] testGuardian() (gas: 57412)
[PASS] testNotGuardian() (gas: 43176)
[PASS] testStorageFuzz(int256) (runs: 256, μ: 38199, ~: 38355)
Suite result: ok. 3 passed; 0 failed; 0 skipped; finished in 114.76ms (84.97ms CPU time)

Ran 22 tests for test/unit/SubnetHardwareRentalBaseTest.t.sol:SubnetHardwareRentalBaseTest
[PASS] testAddHardwareProvider() (gas: 50896)
[PASS] testAddSubnetRentalContract() (gas: 50075)
[PASS] testCannotAddDuplicateSubnetContract() (gas: 48552)
[PASS] testCannotAddZeroAddressSubnetContract() (gas: 20906)
[PASS] testCannotGetExpectedPaymentUSDWithZeroPeriod() (gas: 22816)
[PASS] testCannotRemoveInvalidSubnetContract() (gas: 24733)
[PASS] testGetAvaxUsdPrice() (gas: 24087)
[PASS] testGetExpectedPaymentAVAX() (gas: 35413)
[PASS] testGetExpectedPaymentUSD() (gas: 20960)
[PASS] testOnlyAdminCanSetParameters() (gas: 168875)
[PASS] testRemoveHardwareProvider() (gas: 39838)
[PASS] testRemoveSubnetRentalContract() (gas: 38895)
[PASS] testRentHardwareGGPSwapFailed() (gas: 327075)
[PASS] testRentHardwareInsufficientPayment() (gas: 88976)
[PASS] testRentHardwareInvalidHardwareProvider() (gas: 54548)
[PASS] testRentHardwareWithGGP() (gas: 286817)
[PASS] testSetAvaxPriceFeed() (gas: 29205)
[PASS] testSetPayMargin() (gas: 27228)
[PASS] testSetPaymentCurrency() (gas: 27422)
[PASS] testSetPaymentIncrementUsd() (gas: 27227)
[PASS] testSetPaymentPeriod() (gas: 27250)
[PASS] testSwapAvaxForGGP() (gas: 179432)
Suite result: ok. 22 passed; 0 failed; 0 skipped; finished in 76.96ms (49.80ms CPU time)

Ran 5 tests for test/unit/Delegation.t.sol:DelegationTest
[PASS] testDelegation(uint128) (runs: 256, μ: 491367, ~: 491367)
[PASS] testDelegationDisabled() (gas: 54858)
[PASS] testDelegationIncorrectDeposit(uint128) (runs: 256, μ: 381492, ~: 381492)
[PASS] testDelegationPaused() (gas: 77075)
[PASS] testDelegationZeroRewards(uint128) (runs: 256, μ: 431994, ~: 431994)
Suite result: ok. 5 passed; 0 failed; 0 skipped; finished in 4.20s (4.16s CPU time)

Ran 1 test for test/integration/TestFailure.sol:TestFailure
[PASS] testMinipoolStreamlinerFailure() (gas: 231)
Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 666.02ms (512.17µs CPU time)

Ran 2 tests for
test/integration/ChainHardwareRentalBaseIntegration.t.sol:SubnetHardwareRentalBaseIntegration
[PASS] testCrossSubnetRental() (gas: 329728)
[PASS] testVerifyInitialization() (gas: 52301)
Suite result: ok. 2 passed; 0 failed; 0 skipped; finished in 4.59s (3.55s CPU time)

Ran 10 tests for test/unit/ClaimNodeOp.t.sol:ClaimNodeOpTest
[PASS] testCalculateAndDistributeRewards() (gas: 3256314)
[PASS] testCalculateAndDistributeRewardsInvalidStaker() (gas: 61504)
[PASS] testCalculateAndDistributeRewardsSingleStaker() (gas: 2174748)
[PASS] testCalculateAndDistributeRewardsZeroCycleCount() (gas: 1094980)
[PASS] testClaimAndRestake() (gas: 2147179)
[PASS] testClaimAndRestakePaused() (gas: 2185638)
[PASS] testGetRewardsCycleTotal() (gas: 423768)
[PASS] testIsEligible() (gas: 3705987)
[PASS] testSetRewardsCycleTotal() (gas: 55367)
[PASS] testWhatStatusesNeedMinCollatRatio() (gas: 5435391)
Suite result: ok. 10 passed; 0 failed; 0 skipped; finished in 322.57ms (294.87ms CPU time)

Ran 1 test for test/unit/ERC20Upgradeable.t.sol:ERC20Invariants
[PASS] invariantBalanceSum() (runs: 256, calls: 3840, reverts: 2282)

╭------------+--------------+-------+---------+----------╮
| Contract   | Selector     | Calls | Reverts | Discards |
+========================================================+
| BalanceSum | approve      | 779   | 0       | 0        |
|------------+--------------+-------+---------+----------|
| BalanceSum | burn         | 790   | 778     | 0        |
|------------+--------------+-------+---------+----------|
| BalanceSum | mint         | 773   | 25      | 0        |
|------------+--------------+-------+---------+----------|
| BalanceSum | transfer     | 732   | 722     | 0        |
|------------+--------------+-------+---------+----------|
| BalanceSum | transferFrom | 766   | 757     | 0        |
╰------------+--------------+-------+---------+----------╯

Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 1.14s (1.13s CPU time)

Ran 15 tests for test/unit/MinipoolStreamlinerV2.t.sol:MinipoolStreamlinerV2Test
[PASS] testBatchInvalidHardwareProvider() (gas: 1030497)
Logs:
  miniipoolavax 1000000000000000000000

[PASS] testBatchMismatchedFunds() (gas: 60182)
[PASS] testBatchRelaunchWithNodeRentalAndGGP() (gas: 3533231)
[PASS] testBatchTooManyMinipools() (gas: 96686)
[PASS] testCreateNoNodeRentalNoGGP() (gas: 990959)
[PASS] testCreateNoNodeRentalYesGGP() (gas: 1110532)
[PASS] testCreateNodeRentalNoGGP() (gas: 1033034)
[PASS] testCreateNodeRentalWithGGP() (gas: 1150706)
[PASS] testCreateOrRelaunchStreamlinedMinipoolInvalidHardwareProvider() (gas: 61447)
[PASS] testCreateStakeGGP() (gas: 1050855)
[PASS] testRelaunchNoNodeRentalNoGGP() (gas: 2002861)
[PASS] testRelaunchNoNodeRentalYesGGP() (gas: 2168697)
[PASS] testRelaunchNotOwner() (gas: 2036259)
[PASS] testRelaunchWithNodeRentalAndGGP() (gas: 2140533)
[PASS] testRelaunchWithOonodzNoGGP() (gas: 2018294)
Suite result: ok. 15 passed; 0 failed; 0 skipped; finished in 412.47ms (364.04ms CPU time)

Ran 11 tests for test/unit/MultisigManager.t.sol:MultisigManagerTest
[PASS] testDisableMultisig() (gas: 129915)
[PASS] testDisableMultisigNotFound() (gas: 48985)
[PASS] testEnableMultisig() (gas: 135885)
[PASS] testEnableMultisigNotFound() (gas: 21949)
[PASS] testEnableMultisigNotGuardian() (gas: 18467)
[PASS] testFindActive() (gas: 230123)
[PASS] testMultisigLimit() (gas: 562152)
[PASS] testRegisterMultisig() (gas: 113376)
[PASS] testRegisterMultisigAlreadyRegistered() (gas: 24028)
[PASS] testRegisterMultisigNotGuardian() (gas: 18498)
[PASS] testWithdrawUnclaimedGGP() (gas: 596846)
Suite result: ok. 11 passed; 0 failed; 0 skipped; finished in 44.24ms (18.80ms CPU time)

Ran 4 tests for test/unit/Ocyticus.t.sol:OcyticusTest
[PASS] testAddRemoveDefender() (gas: 50724)
[PASS] testDisableAllMultisigs() (gas: 176472)
[PASS] testOnlyDefender() (gas: 215037)
[PASS] testPauseEverything() (gas: 181611)
Suite result: ok. 4 passed; 0 failed; 0 skipped; finished in 37.83ms (14.24ms CPU time)

Ran 1 test for test/unit/OneInchMock.t.sol:OneInchMockTest
[PASS] testSetRateToEth() (gas: 19335)
Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 76.42ms (372.08µs CPU time)

Ran 2 tests for test/unit/Oracle.t.sol:OracleTest
[PASS] testGGPPriceInAvax() (gas: 92572)
[PASS] testOneInch() (gas: 68016)
Suite result: ok. 2 passed; 0 failed; 0 skipped; finished in 27.52ms (2.10ms CPU time)

Ran 1 test for test/unit/MEVRewards.t.sol:MEVRewardsTest
[PASS] testRate() (gas: 68288)
Logs:
  current rate: 1133113067453039513
  current WAVAX balance: 8671326997166928216244
  WAVAX balance increased to: 8949326997166928216244
  Rate should stay the same 1133113067453039513
  Rate after skipping 14 days: 1134329104921945900

Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 2.24s (1.46s CPU time)
```

```
Ran 43 tests for test/unit/MinipoolManager.t.sol:MinipoolManagerTest
[PASS] testBlskeys() (gas: 1231876)
[PASS] testBondZeroGGP() (gas: 123065)
[PASS] testCalculateGGPSlashAmt() (gas: 85892)
[PASS] testCanClaimAndInitiateStaking() (gas: 1271287)
[PASS] testCancelMinipool() (gas: 1123233)
[PASS] testCancelMinipoolByMultisig() (gas: 1110914)
[PASS] testClaimAndInitiateStaking() (gas: 1452677)
[PASS] testClaimAndInitiateStakingNotEnoughColl() (gas: 2262973)
[PASS] testCreateAndGetMany() (gas: 5828579)
[PASS] testCreateMinipool() (gas: 1659626)
[PASS] testCycleMinipool() (gas: 2202055)
Logs:
  testGas-recordStakingEndAndCycle Gas: 488828

[PASS] testCycleMinipoolCommission() (gas: 1834673)
[PASS] testCycleMinipoolCommissionZero() (gas: 1808614)
[PASS] testCycleMinipoolDurationExceeded() (gas: 1740496)
[PASS] testCycleMinipoolInsufficientAvailableForStaking() (gas: 1879568)
[PASS] testCycleMinipoolInvalidState() (gas: 1895398)
[PASS] testCycleMinipoolUnderCollateralized() (gas: 1831019)
[PASS] testCycleMinipoolZeroGGAVAXReserve() (gas: 1858218)
[PASS] testDurationOutOfBounds() (gas: 1157213)
[PASS] testEmptyState() (gas: 115943)
[PASS] testExpectedRewards() (gas: 64949)
[PASS] testFullCycle_Error() (gas: 1450298)
[PASS] testFullCycle_WithUserFunds() (gas: 1790261)
[PASS] testGasCreateMinipool() (gas: 1025462)
Logs:
  testGasCreateMinipool Gas: 768201

[PASS] testGetMinipool() (gas: 981662)
[PASS] testGetMinipoolCount() (gas: 6303950)
[PASS] testGetMinipools() (gas: 7122980)
[PASS] testGetTotalAVAXLiquidStakerAmt() (gas: 2883039)
[PASS] testMinipoolLaunchEventOnlyOnFirstLaunch() (gas: 2809873)
[PASS] testMultipleCycleNoDelay() (gas: 2616877)
[PASS] testMultipleCycleSmallDelay() (gas: 1898971)
[PASS] testMultipleCycleWithDelay() (gas: 1836077)
[PASS] testOneCycleLongerDuration() (gas: 1835974)
[PASS] testOneCycleWithDelay() (gas: 1814371)
[PASS] testRecordStakingEnd() (gas: 1833144)
[PASS] testRecordStakingEndWithSlash() (gas: 1849988)
[PASS] testRecordStakingEndWithSlashingMoreThanTheyStaked() (gas: 1813106)
[PASS] testRecordStakingError() (gas: 1541256)
[PASS] testRecordStakingStart() (gas: 1553685)
[PASS] testRecordStakingStartInvalidStartTime() (gas: 1293244)
[PASS] testSetBLSKeys() (gas: 1281615)
[PASS] testUndercollateralized() (gas: 442484)
[PASS] testWithdrawMinipoolFunds() (gas: 1655993)
Suite result: ok. 43 passed; 0 failed; 0 skipped; finished in 1.26s (1.23s CPU time)

Ran 14 tests for test/unit/MinipoolStreamliner.t.sol:MinipoolStreamlinerTest
[PASS] testBatchMismatchedFunds() (gas: 57545)
[PASS] testBatchRelaunchWithNodeRentalAndGGP() (gas: 3752400)
[PASS] testBatchTooManyMinipools() (gas: 96882)
[PASS] testCreateNoNodeRentalNoGGP() (gas: 1005471)
[PASS] testCreateNoNodeRentalYesGGP() (gas: 1133419)
[PASS] testCreateNodeRentalNoGGP() (gas: 1213051)
[PASS] testCreateNodeRentalWithGGP() (gas: 1313201)
[PASS] testCreateOrRelaunchStreamlinedMinipoolInvalidSubnetRentalContract() (gas: 88968)
[PASS] testCreateStakeGGP() (gas: 1230936)
[PASS] testRelaunchNoNodeRentalNoGGP() (gas: 2017357)
[PASS] testRelaunchNoNodeRentalYesGGP() (gas: 2184197)
[PASS] testRelaunchNotOwner() (gas: 2295069)
[PASS] testRelaunchWithNodeRentalAndGGP() (gas: 2321834)
[PASS] testRelaunchWithOonodzNoGGP() (gas: 2195403)
Suite result: ok. 14 passed; 0 failed; 0 skipped; finished in 472.21ms (415.36ms CPU time)

Ran 4 tests for test/unit/Timelock.t.sol:TimelockTest
[PASS] testMainnetAssumptions() (gas: 34597)
```

```
[PASS] testTimelockAbort() (gas: 67611)
[PASS] testTimelockChangeProxyAdmin() (gas: 121512)
[PASS] testTimelockUpgrade() (gas: 269400)
Suite result: ok. 4 passed; 0 failed; 0 skipped; finished in 4.03s (3.60s CPU time)

Ran 4 tests for test/unit/TokenGGP.t.sol:TokenGGPTest
[PASS] testConstructorMint() (gas: 913017)
[PASS] testMint() (gas: 115003)
[PASS] testMintMaxReached() (gas: 23189)
[PASS] testMintOnlyRewardsPool() (gas: 16409)
Suite result: ok. 4 passed; 0 failed; 0 skipped; finished in 32.53ms (1.95ms CPU time)

Ran 9 tests for test/unit/upgrade/TokenUpgradeTests.t.sol:TokenUpgradeTests
[PASS] testCannotReinitializeAfterUpgrade() (gas: 9024350)
[PASS] testDeployInitializeAndBadAddress() (gas: 6859364)
[PASS] testDeployTransparentProxy() (gas: 6832853)
[PASS] testDomainSeparatorBetweenVersions() (gas: 6638757)
[PASS] testStorageGapDangerouslySet() (gas: 6806600)
[PASS] testStorageGapSafe() (gas: 6659143)
[PASS] testUpgradeToLatestVersionTimelockScenario() (gas: 7709468)
[PASS] testUpgradeToStAVAXNaming() (gas: 6036638)
[PASS] testUpgradeToStAVAXTimelockScenario() (gas: 6644813)
Suite result: ok. 9 passed; 0 failed; 0 skipped; finished in 64.29ms (30.87ms CPU time)

Ran 29 tests for test/unit/ERC20Upgradeable.t.sol:ERC20UpgradeableTest
[PASS] invariantMetadata() (runs: 256, calls: 3840, reverts: 2719)

╭----------------------+--------------+-------+---------+----------╮
| Contract             | Selector     | Calls | Reverts | Discards |
+==================================================================+
| MockERC20Upgradeable | approve      | 579   | 0       | 0        |
|----------------------+--------------+-------+---------+----------|
| MockERC20Upgradeable | burn         | 555   | 547     | 0        |
|----------------------+--------------+-------+---------+----------|
| MockERC20Upgradeable | init         | 539   | 539     | 0        |
|----------------------+--------------+-------+---------+----------|
| MockERC20Upgradeable | mint         | 534   | 19      | 0        |
|----------------------+--------------+-------+---------+----------|
| MockERC20Upgradeable | permit       | 535   | 535     | 0        |
|----------------------+--------------+-------+---------+----------|
| MockERC20Upgradeable | transfer     | 553   | 543     | 0        |
|----------------------+--------------+-------+---------+----------|
| MockERC20Upgradeable | transferFrom | 545   | 536     | 0        |
╰----------------------+--------------+-------+---------+----------╯

[PASS] testApprove() (gas: 31121)
[PASS] testApprove(address,uint256) (runs: 256, μ: 31299, ~: 31377)
[PASS] testBurn() (gas: 57222)
[PASS] testBurn(address,uint256,uint256) (runs: 256, μ: 59576, ~: 59925)
[PASS] testInfiniteApproveTransferFrom() (gas: 90082)
[PASS] testMetadata(string,string,uint8) (runs: 256, μ: 1013420, ~: 1020783)
[PASS] testMint() (gas: 53891)
[PASS] testMint(address,uint256) (runs: 256, μ: 53726, ~: 54037)
[PASS] testPermit() (gas: 68080)
[PASS] testPermit(uint248,address,uint256,uint256) (runs: 256, μ: 68206, ~: 68440)
[PASS] testRevert_BurnInsufficientBalance(address,uint256,uint256) (runs: 256, μ: 59373, ~: 59529)
[PASS] testRevert_PermitBadDeadline() (gas: 42455)
[PASS] testRevert_PermitBadDeadline(uint248,address,uint256,uint256) (runs: 256, μ: 45148, ~: 45148)
[PASS] testRevert_PermitBadNonce(uint248,address,uint256,uint256,uint256) (runs: 256, μ: 42795, ~: 42795)
[PASS] testRevert_PermitPastDeadline() (gas: 16492)
[PASS] testRevert_PermitPastDeadline(uint248,address,uint256,uint256) (runs: 256, μ: 18504, ~: 18504)
[PASS] testRevert_PermitReplay() (gas: 71897)
[PASS] testRevert_PermitReplay(uint248,address,uint256,uint256) (runs: 256, μ: 72354, ~: 72354)
[PASS] testRevert_TransferFromInsufficientAllowance() (gas: 82131)
[PASS] testRevert_TransferFromInsufficientAllowance(address,uint256,uint256) (runs: 256, μ: 84973, ~:
85207)
[PASS] testRevert_TransferFromInsufficientBalance() (gas: 62717)
[PASS] testRevert_TransferFromInsufficientBalance(address,uint256,uint256) (runs: 256, μ: 65500, ~:
65816)
[PASS] testRevert_TransferInsufficientBalance() (gas: 56402)
[PASS] testRevert_TransferInsufficientBalance(address,uint256,uint256) (runs: 256, μ: 59032, ~: 59499)
[PASS] testTransfer() (gas: 60382)
```

```
[PASS] testTransfer(address,uint256) (runs: 256, μ: 60461, ~: 60617)
[PASS] testTransferFrom() (gas: 84072)
[PASS] testTransferFrom(address,uint256,uint256) (runs: 256, μ: 90591, ~: 93203)
Suite result: ok. 29 passed; 0 failed; 0 skipped; finished in 5.65s (5.65s CPU time)

Ran 45 tests for test/unit/TokenggAVAX.t.sol:TokenggAVAXTest
[PASS] testAmountAvailableForStaking() (gas: 258588)
[PASS] testCurrentErrorOnInsufficientLiquidity() (gas: 381545)
Logs:
  Low level error data length: 4
  0xbb55fd27

[PASS] testDepositAVAXPaused() (gas: 79684)
[PASS] testDepositAdditionalYieldNonZeroFees() (gas: 377920)
[PASS] testDepositAdditionalYieldWithBaseAmount() (gas: 450026)
[PASS] testDepositAdditionalYieldZeroFees() (gas: 295721)
[PASS] testDepositFromStakingInvalid() (gas: 86984)
[PASS] testDepositPause() (gas: 76352)
[PASS] testDepositStakingRewards() (gas: 846390)
[PASS] testDepositYieldIncreasesValue() (gas: 430040)
[PASS] testDepositYieldMultipleSources() (gas: 409049)
[PASS] testDepositYieldWithNonZeroFees() (gas: 335771)
[PASS] testDepositYieldWithZeroFees() (gas: 251990)
[PASS] testDepositYieldZeroAmount() (gas: 93579)
[PASS] testDepositYieldZeroAmountZeroFee() (gas: 65510)
[PASS] testDonateYield() (gas: 205830)
[PASS] testDonateYieldInsufficientShares() (gas: 189242)
[PASS] testDonateYieldZeroShares() (gas: 15837)
[PASS] testMaxDeposit() (gas: 70432)
[PASS] testMaxMint() (gas: 70531)
[PASS] testMaxRedeem() (gas: 258521)
[PASS] testMaxRedeemPaused() (gas: 206594)
[PASS] testMaxWithdraw() (gas: 258832)
[PASS] testMaxWithdrawPaused() (gas: 206609)
[PASS] testMintPause() (gas: 98734)
[PASS] testPreviewDepositPaused() (gas: 72635)
[PASS] testPreviewMintPaused() (gas: 72632)
[PASS] testRedeemAVAXPaused() (gas: 248357)
[PASS] testRedeemPaused() (gas: 241990)
[PASS] testRedeemWhenNoLiquidityAvailable() (gas: 367255)
[PASS] testReinitialization() (gas: 23176)
[PASS] testReserveLowerThanExpected() (gas: 328836)
[PASS] testRevert_RedeemWithdrawAllAssetsMidRewardsCycle() (gas: 284194)
[PASS] testSingleDepositWithdrawAVAX(uint128) (runs: 256, μ: 164264, ~: 164269)
[PASS] testSingleDepositWithdrawWAVAX(uint128) (runs: 256, μ: 181763, ~: 181768)
[PASS] testSingleMintRedeem(uint128) (runs: 256, μ: 162867, ~: 162872)
[PASS] testTokenSetup() (gas: 26745)
[PASS] testWAVAXTransferEnablesRedemption() (gas: 397092)
[PASS] testWithdrawAVAXPaused() (gas: 247479)
[PASS] testWithdrawForMinipoolStaking() (gas: 1582557)
[PASS] testWithdrawForStaking() (gas: 300117)
[PASS] testWithdrawForStakingAmountTooLarge() (gas: 259508)
[PASS] testWithdrawForStakingDisabled() (gas: 47315)
[PASS] testWithdrawForStakingOverdrawn() (gas: 67203)
[PASS] testWithdrawPaused() (gas: 242000)
Suite result: ok. 45 passed; 0 failed; 0 skipped; finished in 2.38s (2.34s CPU time)

Ran 34 tests for test/unit/TokenggAVAXAccessControl.t.sol:TokenggAVAXAccessControlTest
[PASS] testAcceptAdmin() (gas: 74814)
[PASS] testAcceptAdminOnlyPendingAdmin() (gas: 50902)
[PASS] testAcceptAdminRequiresPendingTransfer() (gas: 20431)
[PASS] testAdminCanRevokeOwnRoleWithPending() (gas: 54376)
[PASS] testAdminCannotRevokeOwnRoleWithoutPending() (gas: 25275)
[PASS] testAdminTransferCancelEmitsEvent() (gas: 41030)
[PASS] testAdminTransferEventsEmitted() (gas: 77336)
[PASS] testCancelAdminTransfer() (gas: 46096)
[PASS] testCancelAdminTransferOnlyAdmin() (gas: 55660)
[PASS] testCancelAdminTransferRequiresPendingTransfer() (gas: 22723)
[PASS] testCannotGrantDefaultAdminRole() (gas: 76719)
[PASS] testCompleteAdminTransferFlow() (gas: 143851)
[PASS] testGrantDelegatorRole() (gas: 57709)
[PASS] testGrantRoleEmitsEvent() (gas: 51767)
```

```
[PASS] testGrantRoleIdempotent() (gas: 54364)
[PASS] testGrantRoleOnlyAdmin() (gas: 27792)
[PASS] testGrantWithdrawQueueRole() (gas: 57641)
[PASS] testMultipleTransferAdminOverwritesPending() (gas: 80408)
[PASS] testRedeemAVAXRequiresWithdrawQueueRole() (gas: 23023)
[PASS] testRenounceAdminFunctionRemoved() (gas: 184079)
[PASS] testRevokeRole() (gas: 43088)
[PASS] testRevokeRoleEmitsEvent() (gas: 41592)
[PASS] testRevokeRoleIdempotent() (gas: 29222)
[PASS] testRevokeRoleOnlyAdmin() (gas: 59046)
[PASS] testTransferAdmin() (gas: 58558)
[PASS] testTransferAdminCannotBeSelf() (gas: 20942)
[PASS] testTransferAdminCannotBeZeroAddress() (gas: 20832)
[PASS] testTransferAdminOnlyAdmin() (gas: 25563)
[PASS] testTransferAdminToSelfReverts() (gas: 21280)
[PASS] testTransferAdminToZeroAddressReverts() (gas: 21173)
[PASS] testTransferOverwritesPendingAdmin() (gas: 76778)
[PASS] testWithdrawAVAXRequiresWithdrawQueueRole() (gas: 22946)
[PASS] testWithdrawForStakingRequiresDelegatorRole() (gas: 64404)
[PASS] testWithdrawForStakingWorksWithRole() (gas: 330246)
Suite result: ok. 34 passed; 0 failed; 0 skipped; finished in 43.35ms (17.48ms CPU time)

Ran 3 tests for test/unit/TokenggAVAXFirstDepositor.t.sol:TokenggAVAXTestFirstDepositor
[PASS] testInflationAttackNoDeposit() (gas: 4251362)
[PASS] testInflationAttackWithInitialDeposit() (gas: 4353918)
[PASS] testLargerInflationAttackWithInitialDeposit() (gas: 4353861)
Suite result: ok. 3 passed; 0 failed; 0 skipped; finished in 34.27ms (12.81ms CPU time)

Ran 36 tests for test/unit/TokenpstAVAXTest.t.sol:TokenpstAVAXTest
[PASS] testCannotRecoverUnderlyingAsset() (gas: 22442)
[PASS] testCannotRecoverVaultShares() (gas: 24529)
[PASS] testDeposit() (gas: 267261)
[PASS] testDepositEvent() (gas: 237190)
[PASS] testDepositWAVAX() (gas: 250854)
[PASS] testDepositWhenPaused() (gas: 65198)
[PASS] testDepositWithReceive() (gas: 238291)
[PASS] testDepositZeroAmount() (gas: 41602)
[PASS] testInitializeWithNonERC4626Contract() (gas: 2193406)
[PASS] testInitializeWithZeroVault() (gas: 2148359)
[PASS] testMultipleDeposits() (gas: 316988)
[PASS] testNoggAVAXHoldersBeforeRewards() (gas: 313950)
[PASS] testOneggAVAXHolderBeforeRewards() (gas: 382002)
[PASS] testReceiveWhenPaused() (gas: 58643)
[PASS] testRecoverERC20Safe() (gas: 460567)
[PASS] testRecoverERC20SafeOnlyOwner() (gas: 405007)
[PASS] testRecoverERC20SafeZeroAmount() (gas: 461527)
[PASS] testSetPaused() (gas: 33481)
[PASS] testSetPausedOnlyOwner() (gas: 20607)
[PASS] testSetup() (gas: 33931)
[PASS] testStripYieldDefault() (gas: 379432)
[PASS] testStripYieldGasMeasurement() (gas: 365038)
[PASS] testStripYieldNoYield() (gas: 32154)
[PASS] testStripYieldWhenPaused() (gas: 262115)
[PASS] testStripYieldWithstAVAXHolder() (gas: 385867)
[PASS] testWithdrawEvent() (gas: 260500)
[PASS] testWithdrawInsufficientBalance() (gas: 27670)
[PASS] testWithdrawViaQueue() (gas: 502474)
[PASS] testWithdrawViaQueueEvent() (gas: 488841)
[PASS] testWithdrawViaQueueInsufficientBalance() (gas: 27711)
[PASS] testWithdrawViaQueueMultipleUsers() (gas: 812120)
[PASS] testWithdrawViaQueueWhenPaused() (gas: 265870)
[PASS] testWithdrawViaQueueZeroAmount() (gas: 25574)
[PASS] testWithdrawWhenPaused() (gas: 265814)
[PASS] testWithdrawZeroAmount() (gas: 25461)
[PASS] testWithdrawalAffectsonExcessShares() (gas: 547851)
Suite result: ok. 36 passed; 0 failed; 0 skipped; finished in 79.21ms (52.11ms CPU time)

Ran 1 test for test/unit/upgrade/MinipoolStreamlinerUpgrade.t.sol:MinipoolStreamlinerUpgradeTest
[PASS] testUpgradeMinipoolStreamliner() (gas: 12268730)
Logs:
  0xc7183455a4C133Ae270771860664b6B7ec320bB1
```

```
Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 5.02s (1.64s CPU time)

Ran 2 tests for test/unit/TwapGGP.t.sol:TwapGGPTest
[PASS] test1() (gas: 369952)
Logs:
  blockTimestampLast 1755621479
  price0Average 61993646252789092
  price0CumulativeLast 76774299990558796138117096190423505654958
  GGP Spot 61993646252789092
  GGP Twap 61993646252789092
  ---------
  Skip 1 hour and Swap 1000 WAVAX -> GGP
  GGP Spot 75822646060056207
  GGP Twap 61993646252789092
  Skip 1 hour and Swap 1000 WAVAX -> GGP
  GGP Spot 91040520187324497
  GGP Twap 68908146156422649
  Skip 1 hour and Swap 1000 WAVAX -> GGP
  GGP Spot 107646886523952721
  GGP Twap 76285604166723265
  Skip 1 hour and Swap 1000 WAVAX -> GGP
  GGP Spot 125641395210892972
  GGP Twap 84125924756030629
  Skip 1 hour and Swap all GGP back to wavax
  GGP Spot 62111998786021801
  GGP Twap 92429018847003097
  Skip to end of 24 hour period and update twap checkpoint
  GGP Spot 62111998786021801
  GGP Twap 68428044632059571
  No new trades, skip 1 day and update twap checkpoint
  GGP Spot 62111998786021801
  GGP Twap 62111998786021801

[PASS] testReplaceOneInchMock() (gas: 57450)
Suite result: ok. 2 passed; 0 failed; 0 skipped; finished in 4.26s (2.36s CPU time)

Ran 9 tests for test/unit/Vault.t.sol:VaultTest
[PASS] testAllowedTokens() (gas: 133879)
[PASS] testDepositAvaxFromRegisteredContract() (gas: 125843)
[PASS] testDepositAvaxFromUnRegisteredContract() (gas: 31351)
[PASS] testDepositTokenFromRegisteredContract() (gas: 201293)
[PASS] testDepositTokenFromUnregisteredContract() (gas: 86025)
[PASS] testTransferAvaxFromRegisteredContract() (gas: 125900)
[PASS] testTransferAvaxNotRegisteredContract() (gas: 91669)
[PASS] testTransferTokenFromRegisteredContract() (gas: 179694)
[PASS] testTransferTokenFromUnregisteredContract() (gas: 26968)
Suite result: ok. 9 passed; 0 failed; 0 skipped; finished in 101.40ms (19.68ms CPU time)

Ran 59 tests for test/unit/WithdrawQueue.t.sol:WithdrawQueueTest
[PASS] testAvailableAVAXCalculation() (gas: 1025372)
[PASS] testBatchCancelWithSomeClaimableRequests() (gas: 1191198)
[PASS] testBatchExpiredFundsReclaimedEvent() (gas: 570466)
[PASS] testCancelAfterClaim() (gas: 532048)
[PASS] testCancelFulfilledBeforeClaimable() (gas: 571012)
[PASS] testCancelFulfilledRequest() (gas: 585690)
[PASS] testCancelFulfilledRequestAfterExchangeRateChange() (gas: 661916)
[PASS] testCancelNonExistentRequest() (gas: 25544)
[PASS] testCancelPendingRequestAfterExchangeRateIncrease() (gas: 498552)
[PASS] testCancelPendingRequestSameExchangeRate() (gas: 398059)
[PASS] testCancelRequestNotFound() (gas: 25881)
[PASS] testCancelRequestNotYours() (gas: 447657)
[PASS] testCancelRequestsBatch() (gas: 1468579)
[PASS] testCancelRequestsWithLimit() (gas: 1145524)
[PASS] testCannotCancelAfterClaimable() (gas: 578173)
[PASS] testCannotClaimAfterExpiration() (gas: 572117)
[PASS] testCannotClaimAfterReclaim() (gas: 567602)
[PASS] testClaimUnstake() (gas: 537743)
[PASS] testConfigurableRequestsLimit() (gas: 4283167)
Logs:
  WithdrawQueue.depositFromStaking (fulfilling 10 of 15 requests with limit=10) - Pending remaining: 5
  WithdrawQueue.depositFromStaking (fulfilling 10 of 15 requests with limit=10) - Fulfilled: 10
```

```
[PASS] testDepositAdditionalYield() (gas: 127777)
[PASS] testDepositAdditionalYieldAutoFulfillsPendingRequests() (gas: 591898)
[PASS] testDepositAdditionalYieldWithInsufficientAvailableAVAX() (gas: 591076)
[PASS] testDepositExactAmountForNextRequest() (gas: 965605)
[PASS] testDepositFromStakingMaxRequestsLimit() (gas: 8635998)
Logs:
  WithdrawQueue.depositFromStaking (fulfilling 25 of 30 requests) gas: 2975683
  WithdrawQueue.depositFromStaking (fulfilling remaining 5 requests) gas: 835896

[PASS] testDepositFromStakingWithFeeCalculation() (gas: 856997)
[PASS] testDualSetLifecycleAndCleanup() (gas: 557052)
[PASS] testEnumerableSetHelperFunctions() (gas: 1188847)
[PASS] testFixBoundsChecking() (gas: 477246)
[PASS] testGetExpiredRequestsCountWithPendingRequests() (gas: 995070)
[PASS] testGetRequestsByOwnerPagination() (gas: 3322113)
[PASS] testInitialization() (gas: 28360)
[PASS] testInitializationEvent() (gas: 3652915)
[PASS] testMaxRequestsPerStakingDeposit() (gas: 64302)
[PASS] testMinUnstakeOnBehalfOfAmt() (gas: 64109)
[PASS] testMultipleCancellationsWithRateChanges() (gas: 782734)
[PASS] testMultipleRequestsFromSameUser() (gas: 662114)
[PASS] testMultipleYieldDepositsAutoFulfillQueue() (gas: 1160039)
[PASS] testPartialYieldFulfillment() (gas: 1042010)
[PASS] testReceiveAVAXReverts() (gas: 51489)
[PASS] testReclaimExpiredFunds() (gas: 585461)
[PASS] testReclaimExpiredFundsBeforeExpiration() (gas: 576306)
[PASS] testReclaimExpiredFundsMultiple() (gas: 1005698)
[PASS] testReclaimExpiredFundsWithPendingRequests() (gas: 1093852)
[PASS] testReclaimExpiredPendingRequestEvents() (gas: 495107)
[PASS] testReclaimExpiredPendingRequestOriginalShares() (gas: 497300)
[PASS] testReclaimExpiredRequestInvalidState() (gas: 529144)
[PASS] testReclaimExpiredRequestNotExpired() (gas: 570772)
[PASS] testReclaimExpiredRequestNotFound() (gas: 22483)
[PASS] testReclaimExpiredRequestPending() (gas: 501161)
[PASS] testReentrancyProtectionOnCancelRequest() (gas: 4395838)
[PASS] testRequestUnstakeBasic() (gas: 465085)
[PASS] testRequestUnstakeInsufficientBalance() (gas: 192899)
[PASS] testRequestUnstakeOnBehalfOfExactMinimum() (gas: 453420)
[PASS] testRequestUnstakeOnBehalfOfMinimumMet() (gas: 462151)
[PASS] testRequestUnstakeOnBehalfOfMinimumNotMet() (gas: 164799)
[PASS] testRequestUnstakeZeroShares() (gas: 19599)
[PASS] testSetExpirationDelay() (gas: 64855)
[PASS] testSetUnstakeDelay() (gas: 64879)
[PASS] testcanClaimRequest() (gas: 574744)
Suite result: ok. 59 passed; 0 failed; 0 skipped; finished in 626.78ms (538.88ms CPU time)

Ran 8 tests for test/unit/WithdrawQueueGasTests.t.sol:WithdrawQueueGasTests
[PASS] testGas_CancelRequest_AfterExchangeRateChange() (gas: 443988)
Logs:
  cancelRequest (after exchange rate change) gas: 28273

[PASS] testGas_CancelRequest_Fulfilled() (gas: 640835)
Logs:
  cancelRequest (fulfilled request) gas: 64509

[PASS] testGas_CancelRequest_Pending() (gas: 380248)
Logs:
  cancelRequest (pending request) gas: 19800

[PASS] testGas_CancelRequests_Multiple() (gas: 3479280)
Logs:
  cancelRequests (10 requests, 5 fulfilled) gas: 671939
  Requests cancelled: 10

[PASS] testGas_DepositFromStaking_Staker_NoFee() (gas: 271008)
Logs:
  depositFromStaking (Staker role, 0% fee) gas: 26269

[PASS] testGas_DepositFromStaking_Staker_WithFee() (gas: 355641)
Logs:
  depositFromStaking (Staker role, 10% fee) gas: 110807
```

```
[PASS] testGas_WithdrawQueue_DepositFromStaking_NoRequests() (gas: 297590)
Logs:
  WithdrawQueue.depositFromStaking (no pending requests) gas: 60985

[PASS] testGas_WithdrawQueue_DepositFromStaking_WithRequests() (gas: 1720370)
Logs:
  WithdrawQueue.depositFromStaking (fulfilling 5 requests) gas: 621577

Suite result: ok. 8 passed; 0 failed; 0 skipped; finished in 191.94ms (109.05ms CPU time)

Ran 12 tests for test/invariant/WithdrawQueueInvariants.t.sol:WithdrawQueueInvariants
[PASS] invariant_allocatedFundsAccounting() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+=================================================================================+
| WithdrawQueueHandler | addActor                    | 318   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 288   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 262   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 269   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 264   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 308   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 265   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 263   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 262   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 238   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 257   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 282   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 270   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 294   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_delayConsistency() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+=================================================================================+
| WithdrawQueueHandler | addActor                    | 265   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 269   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 287   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 271   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 265   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 292   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 285   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 290   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 264   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 251   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 288   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 295   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 281   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 237   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_expectedAssetsValidity() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+==================================================================================+
| WithdrawQueueHandler | addActor                    | 282   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 273   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 266   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 271   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 264   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 285   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 297   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 320   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 250   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 275   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 287   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 240   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 266   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 264   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_fifoProcessing() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+==================================================================================+
| WithdrawQueueHandler | addActor                    | 287   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 265   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 310   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 255   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 275   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 252   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 257   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 268   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 277   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 284   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 294   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 288   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 256   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 272   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_fulfilledRequestCompleteness() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+=================================================================================+
| WithdrawQueueHandler | addActor                    | 279   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 263   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 251   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 314   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 261   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 235   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 274   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 266   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 287   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 294   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 269   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 299   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 281   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 267   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_noNegativeBalances() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+=================================================================================+
| WithdrawQueueHandler | addActor                    | 299   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 271   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 318   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 238   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 246   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 294   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 274   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 263   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 286   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 279   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 258   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 278   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 264   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 272   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_queueOrderingConsistency() (runs: 256, calls: 3840, reverts: 0)
```

```
╭----------------------+-------------------------------+-------+---------+----------╮
| Contract             | Selector                      | Calls | Reverts | Discards |
+===================================================================================+
| WithdrawQueueHandler | addActor                      | 297   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                   | 254   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests        | 263   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest                 | 256   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                  | 266   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking            | 277   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX               | 308   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds           | 278   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake                | 253   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards               | 267   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest   | 265   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit         | 299   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest         | 272   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking            | 285   | 0       | 0        |
╰----------------------+-------------------------------+-------+---------+----------╯

[PASS] invariant_requestOwnershipConsistency() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-------------------------------+-------+---------+----------╮
| Contract             | Selector                      | Calls | Reverts | Discards |
+===================================================================================+
| WithdrawQueueHandler | addActor                      | 298   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                   | 281   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests        | 273   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest                 | 311   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                  | 264   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking            | 274   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX               | 298   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds           | 272   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake                | 259   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards               | 258   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest   | 245   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit         | 280   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest         | 288   | 0       | 0        |
|----------------------+-------------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking            | 239   | 0       | 0        |
╰----------------------+-------------------------------+-------+---------+----------╯

[PASS] invariant_requestStateExclusivity() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-------------------------------+-------+---------+----------╮
| Contract             | Selector                      | Calls | Reverts | Discards |
+===================================================================================+
| WithdrawQueueHandler | addActor                    | 276   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 271   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 293   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 267   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 315   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 234   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 276   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 298   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 267   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 282   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 269   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 253   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 271   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 268   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_shareConservation() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+=================================================================================+
| WithdrawQueueHandler | addActor                    | 288   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 257   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 268   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 280   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 262   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 274   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 264   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 271   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 286   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 270   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 274   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 302   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 269   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 275   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_timeProgression() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+=================================================================================+
| WithdrawQueueHandler | addActor                    | 270   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 279   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 269   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 276   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 255   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 282   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 302   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 273   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 273   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 293   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 283   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 262   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 288   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 235   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

[PASS] invariant_totalSystemBalanceConservation() (runs: 256, calls: 3840, reverts: 0)

╭----------------------+-----------------------------+-------+---------+----------╮
| Contract             | Selector                    | Calls | Reverts | Discards |
+=================================================================================+
| WithdrawQueueHandler | addActor                    | 249   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | advanceTime                 | 295   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelMultipleRequests      | 285   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | cancelRequest               | 272   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | claimUnstake                | 273   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositFromStaking          | 275   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | depositToGGAVAX             | 276   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | reclaimExpiredFunds         | 305   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | requestUnstake              | 267   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | simulateRewards             | 266   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testClaimNonExistentRequest | 253   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroAmountDeposit       | 265   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | testZeroSharesRequest       | 287   | 0       | 0        |
|----------------------+-----------------------------+-------+---------+----------|
| WithdrawQueueHandler | withdrawForStaking          | 272   | 0       | 0        |
╰----------------------+-----------------------------+-------+---------+----------╯

Suite result: ok. 12 passed; 0 failed; 0 skipped; finished in 35.18s (57.25s CPU time)

Ran 26 tests for test/unit/ERC4626Std.t.sol:ERC4626StdTest
[PASS] test_RT_deposit_redeem((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 622707,
~: 622994)
[PASS] test_RT_deposit_withdraw((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 622764,
~: 622999)
[PASS] test_RT_mint_redeem((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 626092, ~:
626244)
[PASS] test_RT_mint_withdraw((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 626001, ~:
625715)
[PASS] test_RT_redeem_deposit((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 623697,
~: 623969)
```

```
[PASS] test_RT_redeem_mint((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 623412, ~:
623898)
[PASS] test_RT_withdraw_deposit((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 626370,
~: 627383)
[PASS] test_RT_withdraw_mint((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 626436, ~:
627294)
[PASS] test_asset((address[4],uint256[4],uint256[4],int256)) (runs: 256, μ: 568967, ~: 569096)
[PASS] test_convertToAssets((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 573236, ~:
573470)
[PASS] test_convertToShares((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 573020, ~:
573364)
[PASS] test_deposit((address[4],uint256[4],uint256[4],int256),uint256,uint256) (runs: 256, μ: 617511, ~:
617905)
[PASS] test_maxDeposit((address[4],uint256[4],uint256[4],int256)) (runs: 256, μ: 569699, ~: 570073)
[PASS] test_maxMint((address[4],uint256[4],uint256[4],int256)) (runs: 256, μ: 569857, ~: 570038)
[PASS] test_maxRedeem((address[4],uint256[4],uint256[4],int256)) (runs: 256, μ: 573194, ~: 574444)
[PASS] test_maxWithdraw((address[4],uint256[4],uint256[4],int256)) (runs: 256, μ: 573515, ~: 574452)
[PASS] test_mint((address[4],uint256[4],uint256[4],int256),uint256,uint256) (runs: 256, μ: 620821, ~:
621165)
[PASS] test_previewDeposit((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 612311, ~:
612689)
[PASS] test_previewMint((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 615493, ~:
616065)
[PASS] test_previewRedeem((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 614555, ~:
614754)
[PASS] test_previewWithdraw((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ: 618154, ~:
618711)
[PASS] test_redeem((address[4],uint256[4],uint256[4],int256),uint256,uint256) (runs: 256, μ: 623122, ~:
623584)
[PASS] test_redeem_zero_allowance((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ:
586417, ~: 586375)
[PASS] test_totalAssets((address[4],uint256[4],uint256[4],int256)) (runs: 256, μ: 568862, ~: 569550)
[PASS] test_withdraw((address[4],uint256[4],uint256[4],int256),uint256,uint256) (runs: 256, μ: 626559, ~:
626936)
[PASS] test_withdraw_zero_allowance((address[4],uint256[4],uint256[4],int256),uint256) (runs: 256, μ:
589921, ~: 590286)
Suite result: ok. 26 passed; 0 failed; 0 skipped; finished in 45.61s (99.53s CPU time)

Ran 39 test suites in 51.92s (127.55s CPU time): 534 tests passed, 0 failed, 0 skipped (534 total tests)
```

Changelog

- 2025-08-01 - Initial report
- 2025-08-19 - Final report

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp's team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers 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 500 audits and secured over $200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

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, 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

Timeliness of content

Quantstamp