

Instantly share code, notes, and snippets.

0xJCN / **Audit.md** Secret

Created 2 months ago

[Code](#) [Revisions](#) 1

 **Audit.md**

[Silo V2 4.0.0 Release Audit](#)

Foreword

This audit reviews the changes introduced in the Silo V2 4.0.0 release, with a particular focus on the newly introduced liquidation by defaulting mechanism, fixes to liquidation edge cases, backward compatibility updates for gauges, and the removal of same asset borrowing functionality.

The scope of this engagement is limited to a diff-based review of the contracts modified as part of this release. As such, this report does not attempt to re-establish the full correctness of the protocol, but instead evaluates whether the introduced changes behave as intended, introduce unexpected edge cases, or alter protocol assumptions in ways that may impact users, integrators, or off-chain infrastructure.

Findings Summary

ID	Description	Severity
I-01	Secondary effects of defaulting liquidations	Informational
I-02	Defaulting liquidation can implicitly favor DAO/deployer revenue over lender interest	Informational
I-03	Sophisticated users can miss out on distributed collateral from defaulting liquidation	Informational
I-04	hookReceiverConfig can return deprecated hook actions	Informational

[I-01] Secondary effects of defaulting liquidations

Description

Defaulting liquidation introduces a second liquidation path that can become available before a position is underwater, due to an additional ~2.5% liquidation threshold margin. As a result, both normal liquidation and defaulting liquidation may be valid at a given time for the same unhealthy position.

Unlike normal liquidation, where repaid debt remains in the debt silo and continues to accrue interest, defaulting liquidation writes off debt and compensates lenders via a distribution of the borrower's collateral shares from the opposing silo, along with 80% of the liquidation fee.

Because defaulting liquidation is not strictly limited to situations where bad debt is created, its use introduces several non-obvious secondary effects.

1. Lender exposure shifts from interest-bearing to non-accruing assets

Defaulting liquidation is only supported in one-way markets, where the collateral Silo shares do not accrue interest. When defaulting liquidation occurs, lenders lose part of their interest bearing exposure in the debt silo and gain non interest bearing collateral shares from the opposing Silo (protected assets are prioritized). Note that lenders will also receive an immediate kickback via 80% of the liquidation fee, but this is strictly a short term gain.

To resume earning interest on all of their underlying assets, lenders must claim the distributed shares via the gauge, withdraw the underlying assets, and redeposit them into the debt silo. If this process is delayed, lenders will incur a time-dependent opportunity cost from missed interest accrual.

2. Lower liquidation barrier can change liquidator behavior

Defaulting liquidation significantly lowers participation requirements since no upfront capital and/or flash loan infrastructure is required. These requirements are especially relevant to large positions, where highly sophisticated users are the primary liquidators.

While defaulting liquidators receive only 20% of the liquidation fee, the reduced barrier to entry may incentivize defaulting liquidation even when normal liquidation is feasible and would preserve more long-term value for lenders.

3. Immediate share price deflation impacts integrators

Defaulting liquidation writes off assets without reducing share supply, causing an immediate decrease in the debt silo's share price. This is expected, but notably this can occur even when the position is not underwater. Downstream integrations inherit this effect. For example, Silo vaults aggregate exposure across multiple markets. A defaulting liquidation in a supported market can immediately reduce vault share price.

As a result, vault depositors may experience losses during otherwise normal market conditions, due solely to the liquidation path selected.

Recommendation

Consider documenting these secondary effects so lenders, liquidators, and integrators understand the economic tradeoffs introduced by defaulting liquidation, particularly in scenarios where both liquidation paths are valid.

[I-02] Defaulting liquidation can implicitly favor DAO/deployer revenue over lender interest

Description

During a defaulting liquidation, the collateral in the debt silo is cleared proportionally to the debt being cleared. If the debt cleared exceeds the total collateral in the silo, the excess is subtracted from the `daoAndDeployerRevenue` state variable. This ensures that accrued, but unpaid, fees do not remain tracked in the system and become claimable by the DAO/deployer when future deposits occur.

However, there is a subtle quirk due to how interest accrual works: pending interest is optimistically split between lenders (tracked in `totalAssets[Collateral]`) and the DAO/deployer (tracked by `daoAndDeployerRevenue`). When a portion of this interest is already paid, the protocol does not distinguish between which portion pertains to lenders versus fees. During a defaulting liquidation, clearing the entire collateral means that any paid interest is effectively credited entirely to the DAO/deployer, implicitly favoring the fee recipients over lenders.

Consider a simplified example below:

Initial state

```
total_collateral = 100
total_debt = 100
```

Interest accrues

```
interest_accrued = 10
lender_portion = 5
fee_recipient_portion = 5

total_collateral = 105
total_debt = 110
daoAndDeployerRevenue = 5
```

Portion of interest is paid

```
interest_paid = 5

total_collateral = 105
total_debt = 105
daoAndDeployerRevenue = 5
```

Borrower undergoes defaulting liquidation

```
interest_paid = 5

total_collateral = 0
total_debt = 0
daoAndDeployerRevenue = 5 # DAO/Deployer can claim all paid interest
```

This behavior creates an implicit bias in favor of DAO/deployer revenue at the expense of lender interest, which may not be immediately obvious from the protocol's mechanics.

Recommendation

Document this behavior to ensure users and integrators understand that defaulting liquidation can implicitly prioritize DAO/deployer revenue over lender interest in this edge case.

[I-03] Sophisticated users can miss out on distributed collateral from defaulting liquidation

Description

When a position undergoes a defaulting liquidation, the borrower's collateral shares (equivalent to value of debt cleared + 80% of the liquidation fee) will be distributed to all lenders of that `debt_silo` via the collateral share tokens of the borrower's `collateral_silo`.

This distribution is performed via an immediate distribution in the gauge associated with the collateral share token of the `debt_silo`. So all current holders of this share token will be eligible to claim the distribution. However, sophisticated users may opt to utilize their collateral shares in different protocols, during which they will not own said shares. It is therefore possible for these users to miss out on these distributed shares when a defaulting liquidation occurs.

This is not a direct security issue, but rather an edge case of the distribution mechanism for defaulting liquidations. This edge case becomes more impactful in the scenario where the collateral controller gauge is otherwise not active (no reward distributions announced) and therefore lenders may be incentivized to lend/stake their shares in other protocols. In this case, the reward distribution from a defaulting liquidation may occur unexpectedly to users, which can negatively affect sophisticated lenders who are temporarily not holding their shares.

Recommendation

Consider documenting this edge case so users are aware of the opportunity cost of deploying their collateral shares in other protocols in the event that a defaulting liquidation occurs.

[I-04] `hookReceiverConfig` can return deprecated hook actions

Description

Functionality related to borrowing the same asset as supplied and switching collateral between silos has been deprecated at the protocol level. However, hook configurations can still have the corresponding deprecated hook bits set, and the `hookReceiverConfig` view function may return hook bitmaps that include these deprecated actions.

While this does not pose a security risk, it may confuse integrators or off-chain systems that rely on `hookReceiverConfig` to infer supported protocol functionality.

Recommendation

Consider explicitly documenting that `hookReceiverConfig` may return deprecated hook actions. A potential code change could be to prevent deprecated action bits from being configured for hooks. However, since this issue is limited to observability, such a code change may introduce unnecessary complexity and should be handled with care if implemented.