XRPL Commons — Vote commons-vote-XLS66 Amendment Vote Position — XLS-0066 Lending Protocol

Issued by XRPL Commons · Position Paper

Audited build rippled `release-3.1`, tag `3.1.3 ( 46b241ace8 )`

Specification XLS-0066 Lending Protocol

Date 26 May 2026

Position: Not voting at this time

XRPL Commons is not casting a vote on `featureLendingProtocol` at this time, neither in favour nor against, because the amendment carries a formal dependency on `featureSingleAssetVault`, whose current state holds an unresolved conformance defect; we will revisit whether to vote once the conditions described in this paper are met (see the companion position paper on XLS-0065).

The audit of the Lending implementation itself, reverified against `tag 3.1.3`, concludes that it is technically mature: no exploit-grade vulnerability was found. Our reservation is exclusively driven by the amendment's formal dependency on `featureSingleAssetVault`, whose current state carries an unresolved conformance defect on the Payment-to-pseudo-account path. Activating XLS-0066 over an unpatched XLS-0065 amplifies a latent vulnerability into a permanently active one.

1. Executive summary

The audit covered the nine Lending transactors (`LoanSet`, `LoanPay`, `LoanManage`, `LoanDelete`, `LoanBrokerSet`, `LoanBrokerDelete`, and the three `LoanBrokerCover-*` transactors) along with the ~2,500-line `LendingHelpers` module which concentrates the interest, cover, and payment-split mathematics. The 3.1.1 to 3.1.3 maintenance cycle touched `LoanPay`, `LoanManage`, `LoanBrokerCoverWithdraw`, and the lending invariants block; we reverified every prior finding against the 3.1.3 source.

Severity	Count	Status at 3.1.3
Critical	0	—
High	0	—
Medium	0	—
Low / Info (hardening)	4	L1 still present. L2, L3 still present. L4 partially addressed.
Adversarial hypotheses dismissed	17	None resurfaced under 3.1.3 changes.

None of the seventeen attack hypotheses explored adversarially survived contact with the code, at either tag. Defences observed are systematic: per-transactor authorisation, monotonic state machine (terminal default, impair/unimpair bracketing), fund conservation asserts, atomic Vault accounting, uniform amendment gating, and clean `tec*` rejection codes on every error path.

2. Lending-side hardening items at 3.1.3

These items are not blocking. They are recorded for a subsequent maintenance cycle.

L1 — Time-arithmetic overflow in the anti-default guard of `LoanManage`. Status: still present. Addition of `NextPaymentDueDate + GracePeriod` as raw `uint32` can wrap near the Ripple-epoch year 2106 and let a broker default a loan before the grace period elapses. Not exploitable in the next decade. Trivial fix: saturate the addition.

L2 — Monotonicity of `lsfLoanDefault`. Status: still not invariant-checked. The 3.1.3 invariant rewrite expanded `InvariantCheck` substantially but did not add the monotonicity assertion. Defence-in-depth recommendation persists.

L3 — Accounting identity `Principal + ManagementFee ≤ TotalValue`. Status: still debug-asserted only. Same conclusion as 3.1.1.

L4 — Asymmetric rounding between `LoanSet` and `LoanBrokerCoverWithdraw` on the minimum cover threshold. Status: partially addressed. `LoanBrokerCoverWithdraw` now passes `scale(currentDebtTotal, vaultAsset)` to `roundToAsset` instead of `currentDebtTotal.exponent()`, which fixes the edge case where `currentDebtTotal == 0` would yield `int::lowest`. The asymmetry with `LoanSet` rounding semantics may still warrant normalisation but the original concern is materially reduced.

None of these would, on its own, justify deferring activation.

3. Coupling with XLS-0065 — central point

Twelve coupling surfaces between Lending and Vault were audited. Every Vault write performed from the Lending side is guarded by explicit invariant assertions, atomic decrement-then-check ordering, and proper return-code propagation (`tecLIMIT_EXCEEDED`, `tecINTERNAL`). The Lending side is correctly defensive. This conclusion holds under 3.1.3, including the broader rewrite of `LoanPay` and `LoanManage`.

However, **activating XLS-0066 mechanically amplifies the unresolved XLS-0065 finding F1**.

F1 of the Vault audit (fund-locking via direct Payment to the pseudo-account, persists in 3.1.3) applies to every Vault used as a liquidity source by a `LoanBroker`. An attacker can brick a competitor broker's Vault and freeze the entire portfolio of loans it backs.

F2 of the Vault audit (asymmetric deposit/withdraw accounting in the presence of unrealised loss, persists in 3.1.3) is effectively dormant without XLS-0066 in production, because only Lending transactors can record latent loss. Activating XLS-0066 transforms F2 from a theoretical asymmetry into a persistent user trap and an extractable front-running primitive.

Activating XLS-0066 before resolving the XLS-0065 findings is therefore a net negative for ecosystem safety, irrespective of the technical quality of the Lending implementation itself.

4. The `fixCleanup3_1_3` amendment

The `rippled 3.1.3` release introduces a separate amendment, `fixCleanup3_1_3` (`Supported::yes`, `VoteBehavior::DefaultsYes`). On the Lending side, this amendment changes the share-denominated `VaultWithdraw` flow that the Lending Protocol depends on; the change

improves limit-check accuracy and is favourable. XRPL Commons supports `fixCleanup3_1_3` on its own merits.

The amendment does not however close any of L1, L2, L3 on the Lending side, nor does it address the XLS-0065 findings F1, F2, F4 that underlie our decision not to vote on XLS-0066 at this time. It is therefore necessary but not sufficient.

5. Positive markers in the Lending implementation, confirmed at 3.1.3

Several engineering qualities deserve to be recorded explicitly.

Invariant coverage. Fourteen invariants were identified. The 3.1.3 rewrite of `InvariantCheck` strengthens the broader ledger invariants but leaves the L2/L3 gaps untouched. The path-discipline backstop continues to enforce the missing invariants in practice.

Interest mechanics. Non-negativity is guaranteed by construction: accrual helpers return zero before the relevant payment date. No double-charging. No drift on time-unit changes. Late-payment interest is mutually exclusive with periodic interest by construction.

Payment decomposition. `computePaymentComponents` clamps each delta to zero, resolves any excess in a deterministic order (interest → management fee → principal), and bounds the total by `valueOutstanding`. No dust-based manipulation vector was found, at either tag.

Cover solvency. The minimum-cover threshold is enforced on every broker withdrawal. Cover clawback is upper-bounded by the excess above the minimum and never reaches into the safety buffer.

Default state machine. Terminal, irreversible, and explicitly protected: any `LoanManage` targeting a defaulted loan is rejected at preclaim. The impair / unimpair pair uses the same value snapshot deterministically.

Multi-signature integrity. Borrower counter-signatures are validated against the live `LoanBroker` owner; substitution is not possible.

Absence of oracle dependency. Specification and code are aligned; the entire oracle-manipulation class of DeFi attacks is excluded by design rather than by mitigation.

6. Conditions for XRPL Commons to revisit its vote

XRPL Commons will reconsider whether to vote on XLS-0066 once:

1. the XLS-0065 amendment is in a votable state per the criteria of the companion position paper (F1 patch, F4 patch, public working-group positions on F2 and on the post-`fixCleanup3_1_3` owner clawback semantics, stable corrective release),
2. activation order is publicly coordinated so that the corrected XLS-0065 is activated before XLS-0066. This is mechanically guaranteed by the `featureSingleAssetVault` gating but should nonetheless be part of the communication plan,
3. `fixCleanup3_1_3` is activated jointly with or before `featureSingleAssetVault` and `featureLendingProtocol`,
4. optionally, the L1 to L4 hardening items are integrated in the same release. Their inclusion strengthens confidence but their absence does not block reconsideration.

7. Use of the period during which we are not voting

During the period in which XRPL Commons is not casting a vote, it will:

- support the Ripple team technically on the XLS-0065 corrective work, in particular the one-line F1 patch, in order to shorten the timeline,
- accompany Lending application developers on testnet to validate their integrations against the pre-correction build and surface regressions early,
- prepare user-facing communication on the economic dynamics specific to XLS-0066 (unrealised loss handling, risk distribution between Vault depositors and borrowers).

8. Position in the event of activation before XLS-0065 resolution

If a majority of validators were to activate XLS-0066 before the XLS-0065 conditions are met, XRPL Commons would:

- still not be casting a vote on XLS-0066 at that point,
- continue to support `fixCleanup3_1_3` on its own merits,
- issue a user advisory on the combined operational risks of F1 and F2 in production,

- recommend that `LoanBroker` operators wait for the corrective release before any production deployment.

Contact — `romain@xrpl-commons.org`