

GMX · SECURITY REVIEW

GMX

Token Delisting Procedure.

A security review of Token Delisting Procedure.

CLIENT	PROJECT	AUDITORS	AUDIT WINDOW	FINAL REPORT
GMX	Token Delisting Procedure	Nicholas Chew, Andreas Zottl	May 13 – May 21, 2026	May 21, 2026

§ 00

Contents.

34 pages: overview, confidence ranking, scope, 24 findings, disclaimer, and about Guardian.

§ 01	Overview	P. 03
§ 02	Confidence Ranking	P. 04
§ 03	Scope & Classifications	P. 05
§ 04	Findings	P. 06
	Main Review · 13 findings	P. 08
	Remediation Review · 11 findings	P. 22
§ 05	Disclaimer	P. 33
§ 06	About Guardian	P. 34

§ 01

Overview.

Guardian reviewed Token Delisting Procedure across 2 review rounds. The final report includes 4 High, 12 Medium, 3 Low, 5 Informational findings.

AUDIT FIRM

Guardian

CLIENT

GMX

PROJECT

Token Delisting Procedure

PREPARED BY

Nicholas Chew, Andreas Zottl

AUDIT WINDOW

May 13 – May 21, 2026

FINAL REPORT

May 21, 2026

METHODOLOGY

Smart contract · Manual · Main Review ·
Remediation Review

1.1 Confidence Ranking

3 / 5

Moderate
Confidence

Guardian assigns a Confidence Ranking of 3 out of 5, combining finding severity, remediation outcome, and the reviewer’s assessment of latent risk. The full scale is on the next page.



1.2 Findings at a Glance

SEVERITY	TOTAL	PENDING	ACK	PARTIAL	RESOLVED	FIX REVIEW
Critical	0	0	0	0	0	0
High	4	4	0	0	0	0
Medium	12	6	1	0	0	5
Low	3	1	0	0	0	2
Informational	5	2	0	0	0	3
All severities	24	13	1	0	0	10

§ 02

Confidence Ranking.

Guardian assigns a project-level confidence score using the scale below. The ranking combines finding severity, remediation outcome, and the reviewer's assessment of latent risk.

CONFIDENCE RANKING	DEFINITION AND RECOMMENDATION	RISK PROFILE
5 Very High Confidence	Codebase is mature, clean, and secure. No High or Critical vulnerabilities were found. Follows modern best practices with high test coverage and thoughtful design. Recommendation: code is highly secure at time of audit.	0 High/Critical findings and few Low/Medium severity findings.
4 High Confidence	Code is clean, well-structured, and adheres to best practices. Design patterns are sound and test coverage is strong. Recommendation: suitable for deployment after remediations; consider periodic review with changes.	0-1 High/Critical findings per engagement week and little to no Medium severity findings.
3 Moderate Confidence ASSIGNED	Code is functional but shows gaps in defensive programming or test coverage. Recommendation: remediate the findings and re-review the changes before deployment.	Some High/Critical findings, with various Medium severity findings.
2 Low Confidence	Recurring issues suggest systemic gaps rather than isolated mistakes. Recommendation: remediate, then commission a follow-up review at a frozen commit.	Over 6 High and Critical findings across the engagement.
1 Very Low Confidence	The volume and severity of findings indicate the codebase is not ready for production. Recommendation: substantial rework, then a full re-review.	Over 12 High and Critical findings across the engagement.

§ 03

Scope & Classifications.

Guardian reviewed Token Delisting Procedure.

IN SCOPE

SOP-5: https://www.notion.so/gmxio/Index-Token-Delisting-35e03574745d802a9ce4c4e1e6e24b7a?source=copy_link

SOP-3: https://www.notion.so/gmxio/Market-Delisting-34203574745d8060ae68e30a8297794b?source=copy_link

3.1 Severity Matrix

	IMP · HIGH	IMP · MED	IMP · LOW
LIK · HIGH	Critical	High	Medium
LIK · MED	High	Medium	Low
LIK · LOW	Medium	Low	Low

IMPACT · HIGH

Significant loss of assets, harm to a group of users, or core protocol functionality is disrupted.

IMPACT · MEDIUM

Small amount of funds can be lost, or ancillary functionality is affected. Users may experience reduced or delayed receipt.

IMPACT · LOW

Unexpected behavior in some functionality that is notable but does not meet the criteria for higher severity.

LIKELIHOOD · HIGH

Possible with reasonable assumptions that mimic on-chain conditions; cost is low compared to amount gained.

LIKELIHOOD · MEDIUM

Possible only in uncommon cases or requires a large amount of capital relative to disruption caused.

LIKELIHOOD · LOW

Unlikely to ever occur in production.

§ 04

Findings.

Grouped by round. Visible IDs reset by severity within each round, so a finding is named the same way in the report as it was in the channel it was written in. Status is where each finding ended up; its own page says how.

ID	TITLE	CATEGORY	SEVERITY	STATUS	PAGE
MAIN REVIEW					
H-01	Full Impact Pool Withdrawal Step Will Likely DoS Itself	DoS	HIGH	PENDING	p. 8
H-02	Runbook Script Commands Do Not Match Actual Script Inputs	Configuration	HIGH	PENDING	p. 9
H-03	Phase 2 Skip Borrowing-Fee Toggle Affects All Markets	Configuration	HIGH	PENDING	p. 10
H-04	Zero-valued config changes skipped by script	Error	HIGH	PENDING	p. 11
M-01	Static Prices Unusable When Ref Price Diverge	DoS	MEDIUM	PENDING	p. 12
M-02	Wrong Role Definitions	Access Control	MEDIUM	PENDING	p. 14
M-03	GLV Market Removal DoS Griefing Attack	DoS	MEDIUM	PENDING	p. 15
M-04	Static Price Fallback Uses Incorrect DataStore Key Shape	Configuration	MEDIUM	PENDING	p. 16
M-05	Phase 2 Misstates How to Disable Borrowing Kink Model	Configuration	MEDIUM	PENDING	p. 17
M-06	LPs Can Game Remaining Delisting Impact Pool Distribution	Gaming	MEDIUM	PENDING	p. 18
L-01	Residual Swap Impact Pool Funds May Be Stuck After Market Disable	Rewards	LOW	PENDING	p. 19
I-01	Swaps Remain Open	Informational	INFORMATIONAL	PENDING	p. 20
I-02	Disabled Markets May Leave Stale Swap Virtual Inventory	Informational	INFORMATIONAL	PENDING	p. 21
REMEDIATION REVIEW					
M-01	Runbook Incorrectly States priceFeedKey Cannot Be Cleared	DoS	MEDIUM	ACKNOWLEDGED	p. 22
M-02	Phase 3 Role Guidance Incorrectly Allows LIMITED_CONFIG_KEEPER	DoS	MEDIUM	REPORTED	p. 23
M-03	Wrong Allowance Spender for Lent Impact Repayment	Configuration	MEDIUM	REPORTED	p. 24

Findings continued.

ID	TITLE	CATEGORY	SEVERITY	STATUS	PAGE
M-04	GLV Removal Before Impact-Pool Distribution Can Misallocate Value	Rewards	MEDIUM	REPORTED	p. 25
M-05	Borrowing Incentive Guidance Can Raise an Ignored Parameter	Configuration	MEDIUM	REPORTED	p. 26
M-06	Residual Long/Short Pool Tokens Are Not Handled	Documentation	MEDIUM	REPORTED	p. 27
L-01	Phase 1c GLV shift-out can be blocked	DoS	LOW	REPORTED	p. 28
L-02	Pending Target-Market Orders Can Revert After Final Disable	DoS	LOW	REPORTED	p. 29
I-01	Process Summary Retains Stale Bilateral Borrowing Reference	Documentation	INFORMATIONAL	REPORTED	p. 30
I-02	Phase 3 Static Price Step References Removed Function	Configuration	INFORMATIONAL	REPORTED	p. 31
I-03	Overcollateralized Small Position Can Block 100% Liquidation Cleanup	DoS	INFORMATIONAL	REPORTED	p. 32

HIGH **H-01**

PENDING

Full Impact Pool Withdrawal Step Will Likely DoS Itself

CATEGORY DoS
REVIEW Main Review

— DESCRIPTION

In Phase 4 the remaining position impact pool is withdrawn. This is a 2-Step flow with a time delay and the `withdrawFromPositionImpactPool.ts` script is used to do so.

This script is called with `FULL_WITHDRAWAL=true` for both creating the action and executing it. Therefore it will try to withdraw everything in the pool that's available at the exact time the script is called.

If during creation and execution of this action the `positionImpactPoolDistributionRate` is positive (which is the case after Phase 1b) then this value will no longer be correct at execution time and a wrong signature is created as the `FULL_WITHDRAWAL=true` flag is used during execution and even if the right signature is created the contract would revert as not enough funds are in the pool.

— RECOMMENDATION

Set the `positionImpactPoolDistributionRate` of the market to 0 before calling the `withdrawFromPositionImpactPool.ts` script.

— RESOLUTION

Status: Pending.

HIGH **H-02**

PENDING

Runbook Script Commands Do Not Match Actual Script Inputs

CATEGORY Configuration

REVIEW Main Review

— DESCRIPTION

Several runbook commands pass environment variables or describe script behavior that is not implemented by the referenced scripts. This can cause operators to believe they are applying targeted delisting actions while the script either ignores the input, applies changes more broadly than expected, or uses hardcoded/default behavior, which could have dangerous unexpected outcomes.+

For example:

- `withdrawFromPositionImpactPool.ts` ignores `MARKET` ; it uses a hardcoded market instead.
- `withdrawFromPositionImpactPool.ts` ignores `RECEIVER` ; it uses a hardcoded receiver address.
- `updateMarketPositionImpactDistributionConfig.ts` ignores `MARKET` ; it applies all pending impact-distribution config changes across configured markets.
- `updateMarketConfig.ts` ignores `SKIP_BORROWING_FEE_FOR_SMALLER_SIDE`

— RECOMMENDATION

Update the runbook and scripts so every documented environment variable is either implemented or removed.

— RESOLUTION

Status: Pending.

HIGH **H-03**

PENDING

Phase 2 Skip Borrowing-Fee Toggle Affects All Markets

CATEGORY Configuration

REVIEW Main Review

— DESCRIPTION

In phase 2 `SKIP_BORROWING_FEE_FOR_SMALLER_SIDE` is update to false as part of the target market's delisting incentives. However, this setting is a global DataStore key, not a market-scoped parameter. As a result, changing this flag makes the smaller open-interest side pay borrowing fees across every market, not only the market being delisted. This can change position economics for unrelated traders and may incentivize liquidity or position exits from unaffected GMX markets.

A even worse impact regarding that is a stepwise drop in the health factor of many positions, as the system does not automatically update borrowing of all the markets before changing this parameter. Therefore the more stale a market's cumulative borrowing factor is (the longer it is untouched) the more pressure is put up on all the positions of the smaller side with a stepwise drop. This can lead to many positions being suddenly liquidatable.

— RECOMMENDATION

Consider one of the following actions:

- Remove this step from the runbook
- Update this parameter so that it acts per market and also consider to always accrue funding before updating it (to avoid the mentioned step wise drops)
- Document this behavior and analyze the market before updating this param to handle it with care

— RESOLUTION

Status: Pending.

HIGH **H-04**

PENDING

Zero-valued config changes skipped by script

CATEGORY Error
REVIEW Main Review

DESCRIPTION

In the market delisting playbook, Phase 1d says set `maxPoolUsdForDeposit` values to 0, and Phase 2 says set `ABOVE_OPTIMAL_USAGE_BORROWING_FACTOR` to 0. However, in `updateMarketConfigUtils.ts` (line 112) the `addConfigItem` returns early for falsy values, so 0 is never added to the multicall.

As a result, the operator runs the script, it returns 0 (the exit code, not the config value), the changeset PR is merged, and the market remains open for deposits and unchanged on the borrowing curve.

RECOMMENDATION

Change `addConfigItem` in `updateMarketConfigUtils.ts` so it only skips truly missing values, not valid zero values:

```
``solidity
if (value === undefined || value === null) {
return;
}
``
```

RESOLUTION

Status: Pending.

MEDIUM **M-01**

PENDING

Static Prices Unusable When Ref Price Diverge

CATEGORY DoS
REVIEW Main Review

DESCRIPTION

`StaticOracleProvider` is intended to support frozen pricing for markets that are being closed. However, static prices are still validated against the live Chainlink reference price when a reference feed exists for the token.

`StaticOracleProvider` explicitly reports that it is not a Chainlink on-chain provider:

```
``plain text
function isChainlinkOnChainProvider() external pure returns (bool) {
return false;
}
`
```

As a result, `Oracle._validatePrices` applies the reference-price deviation check to static prices:

```
``plain text
if (!provider.isChainlinkOnChainProvider()) {
(bool hasRefPrice, uint256 refPrice) = ChainlinkPriceFeedUtils.getPriceFeedPrice(dataStore, token);

if (hasRefPrice) {
_validateRefPrice(token, validatedPrice.min, refPrice, maxRefPriceDeviationFactor);
_validateRefPrice(token, validatedPrice.max, refPrice, maxRefPriceDeviationFactor);
}
}
`
```

This means a frozen token price remains executable only while it stays within `MAX_ORACLE_REF_PRICE_DEVIATION_FACTOR` of the current Chainlink feed. If the live Chainlink price moves far enough away from the frozen price, every non-atomic execution requiring that static token price reverts with `MaxRefPriceDeviationExceeded`. This can leave positions and liquidity stuck or force governance/operators to change oracle configuration again under

M-01 Static Prices Unusable When Ref Price Diverge

CONTINUED

emergency conditions.

This will also lead to reverts if the reason behind shutting down the market is a deprecated chainlink feed as in that case calling `lastRoundData` on the chainlink feed will lead to a revert.

— RECOMMENDATION

Do not apply the live Chainlink reference-price deviation check to explicitly frozen static prices.

— RESOLUTION

Status: Pending.

MEDIUM **M-02**

PENDING

Wrong Role Defintions

CATEGORY Access Control

REVIEW Main Review

DESCRIPTION

The defined roles that are needed to execute certain actions are wrong in multiple places, as it often says that an action can also be executed by the `LIMITED_CONFIG_KEEPER` but in reality it's only executable by the `CONFIG_KEEPER`, this is for example the case for the following actions:

- updating the position impact pool distribution rate in Phase 1b
- Borrowing parameter changes in Phase 2
- Raising min collateral factor for liquidation in Phase 3
- Final market disable

RECOMMENDATION

Recheck the roles in the runbook and update them accordingly to avoid running into access control issues.

RESOLUTION

Status: Pending.

MEDIUM **M-03**

PENDING

GLV Market Removal DoS Griefing Attack

CATEGORY DoS
REVIEW Main Review

DESCRIPTION

During Phase 1c, the runbook requires shifting all GM balance out of the affected market and then calling `removeMarketFromGlv`. Removal requires the GLV market to be disabled and the GLV's accounted GM balance for that market to be exactly zero.

A malicious actor can DoS this process by transferring a dust amount of GM tokens to the GLV market so that zero balance check reverts. The runbook would then require to repeat this process, but the attacker could front run again:

1. GLV market is disabled
2. Keeper creates a GLV shift for the full accounted GM balance
3. Attacker transfers 1 wei GM directly to the GLV before `executeGlvShift`
4. `executeGlvShift` transfers out the accounted GM amount
5. `StrictBank._afterTransferOut` syncs `tokenBalances[market]` to the actual ERC20 balance, which is now the attacker's 1 wei dust
6. `removeMarketFromGlv` reverts with `GlvNonZeroMarketBalance`

RECOMMENDATION

The safest practice would be to add a `CONFIG_KEEPER`-only disabled-market cleanup function that atomically shifts any remaining GM and removes the market from the GLV in the same transaction.

RESOLUTION

Status: Pending.

MEDIUM **M-04**

PENDING

Static Price Fallback Uses Incorrect DataStore Key Shape

CATEGORY Configuration

REVIEW Main Review

— DESCRIPTION

Phase 3 describes a fallback for cases where the token price feed is completely unavailable:

```
dataStore.setUintArray(Keys.staticOraclePriceKey(token), [min, max])
```

This does not match the implementation. `StaticOracleProvider` reads two separate uint values:

```
Keys.staticOraclePriceKey(token, false) // min
```

```
Keys.staticOraclePriceKey(token, true) // max
```

The runbook also suggests switching the oracle provider via `scripts/updateOracleConfigForTokens.ts`, but that script only supports chainlink feeds.

— RECOMMENDATION

Update the fallback instructions to match the actual storage layout and tooling.

— RESOLUTION

Status: Pending.

MEDIUM **M-05**

PENDING

Phase 2 Misstates How to Disable Borrowing Kink Model

CATEGORY Configuration

REVIEW Main Review

— DESCRIPTION

Phase 2 says to “disable kink model” by setting `ABOVE_OPTIMAL_USAGE_BORROWING_FACTOR` to 0. This does not disable the kink model, the kink branch is selected whenever `OPTIMAL_USAGE_FACTOR != 0`.

`ABOVE_OPTIMAL_USAGE_BORROWING_FACTOR` only controls the extra borrowing-rate slope after utilization exceeds `OPTIMAL_USAGE_FACTOR`. Setting it to 0 removes that above-optimal surcharge, but the rate still uses the kink-model base formula

— RECOMMENDATION

Clarify the intended behavior and update the runbook accordingly:

- If the sentence is correct and the goal is to disable the kink model entirely, set `OPTIMAL_USAGE_FACTOR` to 0 for both long and short sides.
- If the goal is only to remove the extra above-optimal surcharge, rewrite the sentence to say that explicitly.

— RESOLUTION

Status: Pending.

MEDIUM **M-06**

PENDING

LPs Can Game Remaining Delisting Impact Pool Distribution

CATEGORY Gaming
REVIEW Main Review

— DESCRIPTION

In the market delisting playbook, Phase 1b starts draining the position impact pool by setting `minPositionImpactPoolAmount = 0` and increasing `positionImpactPoolDistributionRate`. Phase 1d freezes deposits later by setting `MAX_POOL_USD_FOR_DEPOSIT = 0`.

This ordering leaves a window where public deposits are still enabled while the remaining impact-pool distribution is known and scheduled.

An opportunistic LP can deposit between Phase 1b and Phase 1d and capture a pro-rata share of the remaining accelerated impact-pool release. Existing LPs are diluted because their share of the remaining scheduled distribution decreases after the new GM is minted.

This creates an avoidable value transfer during delisting.

— RECOMMENDATION

Add any required treasury liquidity, then freeze public deposits before starting accelerated impact-pool distribution.

— RESOLUTION

Status: Pending.

LOW **L-01**

PENDING

Residual Swap Impact Pool Funds May Be Stuck After Market Disable

CATEGORY Rewards
REVIEW Main Review

— DESCRIPTION

The market delisting runbook withdraws the position impact pool but does not clear the `swapImpactPoolAmount`. If this value is positive when the market is finally disabled, normal swaps can no longer consume it because `validateSwapMarket` requires the market to be enabled.

— RECOMMENDATION

Be aware about that and consider to implement a withdraw from swap impact pool function if needed.

— RESOLUTION

Status: Pending.

INFORMATIONAL I-01

PENDING

Swaps Remain Open

CATEGORY Informational

REVIEW Main Review

— DESCRIPTION

Deposits and OI is frozen during this flow but swaps can still be performed.

— RECOMMENDATION

Think about if this is intended and if not consider to disable swaps with this market during the flow.

— RESOLUTION

Status: Pending.

INFORMATIONAL I-02

PENDING

Disabled Markets May Leave Stale Swap Virtual Inventory

CATEGORY Informational

REVIEW Main Review

— DESCRIPTION

During market delisting, the final disable step requires open interest to be zero, but does not require the market's pool balances to be zero. Since swap virtual inventory is derived from pool amount updates, a disabled market with residual long/short pool balances may continue to influence swap price impact for active markets sharing the same virtual market ID.

— RECOMMENDATION

Be aware about that and consider to add a note in the runbook to check that this value will not be too big to have any real influence on other markets.

— RESOLUTION

Status: Pending.

MEDIUM **M-01**

ACKNOWLEDGED

Runbook Incorrectly States priceFeedKey Cannot Be Cleared

CATEGORY	DoS
LOCATION	Market Delisting Runbook
REVIEW	Remediation Review

— DESCRIPTION

The Market Delisting runbook states for Phase 3 (Option 1) that on-chain cleanup of `priceFeedKey(token)` has no current `Config` / `TimelockConfig` path and would require a contract upgrade.

This is inaccurate for `priceFeedKey`. `Config.setAddress` cannot clear it because `PRICE_FEED` is not allowlisted in `Config._initAllowedBaseKeys`, but `TimelockConfig.signalSetPriceFeed` schedules a direct `DataStore.setAddress(Keys.priceFeedKey(token), priceFeed)` call and does not reject `address(0)`.

Clearing `priceFeedKey(token)` matters because `ChainlinkPriceFeedUtils.getPriceFeedPrice` returns `(false, 0)` when the configured feed address is zero. This skips Chainlink reference-price validation, which otherwise can make `StaticOracleProvider` unusable if the live feed diverges, becomes stale, or reverts.

This should be distinguished from `oracleProviderForToken`: there is no safe dedicated clear path for that key. `Config.setOracleProviderForToken` requires the provider to be enabled, and enabling `address(0)` as an oracle provider would be an unsafe abuse of the provider allowlist.

— RECOMMENDATION

Update the runbooks to state:

- `Config.setAddress` cannot clear `priceFeedKey(token)`.
- `TimelockConfig.signalSetPriceFeed(token, address(0), ...)` can clear `priceFeedKey(token)`.

— RESOLUTION

Status: Acknowledged.

MEDIUM **M-02**

Phase 3 Role Guidance Incorrectly Allows LIMITED_CONFIG_KEEPER

CATEGORY	DoS
LOCATION	Market Delisting Runbook
REVIEW	Remediation Review

— DESCRIPTION

The market delisting runbook correctly states in its prerequisites that Phase 3 requires `CONFIG_KEEPER`, but the Phase 3 forced-liquidation section later says raising `minCollateralFactorForLiquidation` can be executed by `LIMITED_CONFIG_KEEPER` or `CONFIG_KEEPER`.

This regresses the earlier M-02 role-fix finding. `MIN_COLLATERAL_FACTOR_FOR_LIQUIDATION` is included in `Config._initAllowedBaseKeys`, but it is not included in `Config._initAllowedLimitedBaseKeys`. A `LIMITED_CONFIG_KEEPER` call through `Config.setUint` would therefore revert with `InvalidBaseKey`. Operators may attempt the Phase 3 liquidation-threshold update with the wrong signer, causing an avoidable failed transaction or delaying forced cleanup during a time-sensitive delisting.

— RECOMMENDATION

Update Phase 3 to state that changing `minCollateralFactorForLiquidation` requires `CONFIG_KEEPER` only. Keep the prerequisite and per-step role guidance consistent.

— RESOLUTION

Status: Reported during the fix review period.

MEDIUM **M-03**

Wrong Allowance Spender for Lent Impact Repayment

CATEGORY	Configuration
LOCATION	Market Delisting Runbook
REVIEW	Remediation Review

— DESCRIPTION

Phase 4a instructs `fundingAccount` to approve the market contract for both `market.longToken` and `market.shortToken` before executing `reduceLentImpactAmount`.

This is the wrong spender. The timelock executes `ConfigTimelockController.reduceLentImpactAmount`, which calls `PositionImpactPoolUtils.reduceLentAmount`. The token pulls are performed by the executing controller context:

```
``solidity
IERC20(marketProps.longToken).safeTransferFrom(fundingAccount, market, longTokenAmount);
IERC20(marketProps.shortToken).safeTransferFrom(fundingAccount, market, shortTokenAmount);
`
```

The existing test approves `configTimelockController`, not the market token, before executing the repayment.

If operators follow the runbook and approve the market contract, the timelock execution will revert due to missing allowance for `ConfigTimelockController`. This can delay final cleanup and may be especially problematic if the index token's oracle feed is close to expiry or outage.

— RECOMMENDATION

Update Phase 4a to instruct `fundingAccount` to approve `ConfigTimelockController` as the spender for both long and short tokens. If `longToken == shortToken`, approve enough for the combined long-token and short-token transfer amount.

— RESOLUTION

Status: Reported during the fix review period.

MEDIUM **M-04**

GLV Removal Before Impact-Pool Distribution Can Misallocate Value

CATEGORY Rewards

REVIEW Remediation Review

— DESCRIPTION

The runbook shifts/removes the affected market from GLV before starting accelerated position impact-pool distribution. This can transfer value away from GLV LPs. GM pool value subtracts the undistributed position impact pool from market value, so a GLV shift-out values the GLV's GM balance before that impact-pool value has been released. Later, when the impact pool is distributed, the GM value increase accrues only to holders that still own the affected market's GM tokens.

— RECOMMENDATION

Document this behavior or consider to distribute the impact pool twice for a more fair GLV wind down.

— RESOLUTION

Status: Reported during the fix review period.

MEDIUM **M-05**

Borrowing Incentive Guidance Can Raise an Ignored Parameter

CATEGORY Configuration

REVIEW Remediation Review

— DESCRIPTION

The runbook states out to raise the `BORROWING_FACTOR` and/or `BASE_BORROWING_FACTOR`, then set the `OPTIMAL_USAGE_FACTOR` to zero to disable the kink model.

However once `OPTIMAL_USAGE_FACTOR` is zero, the non-kink borrowing calculation is used and `BASE_BORROWING_FACTOR` is ignored. Therefore if the `BASE_BORROWING_FACTOR` is raised, the intended voluntary-exit incentive do not increase.

— RECOMMENDATION

Increase `BORROWING_FACTOR` to the intended effective rate for both long and short sides. The `BASE_BORROWING_FACTOR` only applies while the kink model remains active.

— RESOLUTION

Status: Reported during the fix review period.

MEDIUM **M-06**

Residual Long/Short Pool Tokens Are Not Handled

CATEGORY	Documentation
LOCATION	Market Delisting Runbook
REVIEW	Remediation Review

— DESCRIPTION

In Phase 4e, residual LPs can redeem off-chain by sending GM tokens to the treasury, with the treasury paying them in underlying assets. It then says the treasury-held GM is retained for future handling, such as "admin redemption after a protocol upgrade". The runbook does not include a step that transfers the remaining `longToken` / `shortToken` pool balances from the market to treasury.

As a result, treasury may compensate residual LPs using separate treasury funds while the corresponding `longToken` / `shortToken` backing remains in the disabled market. If residual GM balances are non-trivial, the delisting process can leave material pool assets trapped.

This makes Phase 4e operationally ambiguous: operators can complete the user-facing treasury swap, but the runbook does not specify how to settle the associated residual pool assets that Phase 0 instructed them to track.

— RECOMMENDATION

Document a reviewed timelock-controller batch procedure for residual recovery, including how to calculate transfer amounts, whether to burn treasury-held GM, and whether to update `poolAmountKey` values after disable.

Recovery appears possible through privileged governance operations. `ConfigTimelockController` is granted `CONTROLLER`, and governance can schedule arbitrary calls directly through the underlying timelock controller, via a scheduled batch call

— RESOLUTION

Status: Reported during the fix review period.

LOW **L-01**

Phase 1c GLV shift-out can be blocked

CATEGORY DoS
REVIEW Remediation Review

— DESCRIPTION

Phase 1B of the market delisting playbook says "Multiple shift transactions may be required depending on balance size (choose different `TO_MARKET` values)." That's correct but only names one of four reasons a `MARKET_TOKEN_AMOUNT=ALL` shift can revert. The four constraints are:

1. `GlvMaxMarketTokenBalance{Amount,Usd}Exceeded` : Full amount pushes the destination past its per-GLV cap (this is covered in playbook)
2. `GlvShiftMaxLossExceeded` : Combined withdraw + deposit price impact exceeds `glvShiftMaxLossFactor`
3. `GlvShiftIntervalNotYetPassed` : Wall-clock rate limit on shifts per GLV
4. `MaxPnlFactorExceeded` : Destination's pending PnL near its deposit cap

— RECOMMENDATION

Document these behavior in the runbook, be aware and consider updating the config values (e.g. max loss factor) if needed.

— RESOLUTION

Status: Reported during the fix review period.

LOW L-02

Pending Target-Market Orders Can Revert After Final Disable

CATEGORY DoS

REVIEW Remediation Review

— DESCRIPTION

The market delisting playbook reaches final disable in Phase 4d by setting `isDisabled: true` for the target market. The playbook requires open interest to be zero before Phase 4, but it does not require the target market's pending order queues to be empty.

This leaves an execution gap for already-created target-market orders, including frozen orders. In the guided delist fuzz repro, a frozen SOL-market order remained queued when the SOL market was disabled. A later generic `ExecuteOrder` selected that order and reverted with

```
DisabledMarket(0x324AA6331C60D0089e06c8Ec26112F339a56F398)
```

The index-token delisting runbook notes that pending orders can be cancelled by users after terminal disable, because `cancelOrder` does not need the oracle. Until those orders are cancelled, execution attempts against the stale queue can continue to fail.

— RECOMMENDATION

Add a pending-order drain before Phase 4d final disable.

— RESOLUTION

Status: Reported during the fix review period.

INFORMATIONAL I-01

Process Summary Retains Stale Bilateral Borrowing Reference

CATEGORY	Documentation
LOCATION	Market Delisting Runbook
REVIEW	Remediation Review

— DESCRIPTION

The market delisting v2 runbook removed the Phase 2 instruction to change `SKIP_BORROWING_FEE_FOR_SMALLER_SIDE`, resolving H-03's concern that this is a global setting affecting all markets.

However, the Process Summary still describes Phase 2 as using "high borrowing fees, bilateral charges, disable kink". The phrase "bilateral charges" can lead operators to infer that the global bilateral borrowing toggle should still be changed during delisting.

— RECOMMENDATION

Remove "bilateral charges" from the Process Summary so it matches the corrected Phase 2 body.

— RESOLUTION

Status: Reported during the fix review period.

INFORMATIONAL I-02

Phase 3 Static Price Step References Removed Function

CATEGORY	Configuration
LOCATION	Market Delisting Runbook
REVIEW	Remediation Review

— DESCRIPTION

The market delisting runbook instructs operators to call:

```
`solidity
Config.setStaticPriceForToken(token, pricesParams)
```

The latest v2.2c commit `fd095157` does not expose this function. Static price freezing in that commit is performed through:

```
`solidity
Config.setStaticPriceForMarketIndexToken(market, pricesParams)
```

This function accepts the market token address, loads the market, verifies it exists, derives the `indexToken`, stores the current oracle min/max as static prices, and switches that index token to `StaticOracleProvider`.`

— RECOMMENDATION

Determine which commit the runbook should reference, and update the Phase 3 reference accordingly.

— RESOLUTION

Status: Reported during the fix review period.

INFORMATIONAL I-03

Overcollateralized Small Position Can Block 100% Liquidation Cleanup

CATEGORY DoS

REVIEW Remediation Review

— DESCRIPTION

The runbook relies on setting the `minCollateralFactorForLiquidation` to 100% to make all remaining positions liquidatable. However this is not the case for overcollateralized position. This enables an attack vector to DoS / slow down the delisting runbook as a malicious actor could create a very small, but heavily overcollateralized position before the delisting flow starts.

— RECOMMENDATION

Document this behavior and consider to increase the factor further in that case.

— RESOLUTION

Status: Reported during the fix review period.

§ 05

Disclaimer.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Guardian to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Guardian's position is that each company and individual are responsible for their own due diligence and continuous security. Guardian's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by Guardian is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis.

Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

Notice that smart contracts deployed on the blockchain are not resistant from internal/external exploit. Notice that active smart contract owner privileges constitute an elevated impact to any smart contract's safety and security. Therefore, Guardian does not guarantee the explicit security of the audited smart contract, regardless of the verdict.

§ 06

About Guardian.

Guardian is a smart contract security firm. Every engagement is worked by two independent teams competing on the same scope, each with its own tooling and methods, so no single reviewer's blind spot survives.

\$45B+DIGITAL ASSETS UNDER
REVIEW**200+**

ENGAGEMENTS COMPLETED

500+CRITICAL & HIGH FINDINGS
REPORTED

Reach us at audits@guardianaudits.com.