GUARDIAN AUDITS

SMART CONTRACT SECURITY AUDIT OF

GMX

Summary

Audit Firm Guardian

Prepared By Owen Thurm, Daniel Gelfand, 0xKato

Client Firm GMX

Final Report Date December 17, 2023

Audit Summary

GMX engaged Guardian to review the security of its RewardRouterV2 and Governance updates. From the 5th of December to the 12th of December a team of 3 auditors reviewed the source code in scope. All findings have been recorded in the following report.

Notice that the examined smart contracts are not resistant to internal exploit. For a detailed understanding of risk severity, source code vulnerability, and potential attack vectors, refer to the complete audit report below.

🔗 Blockchain network: **Arbitrum, Avalanche**

✅ Verify the authenticity of this report on Guardian's GitHub: https://github.com/guardianaudits

📊 Code coverage & PoC test suite: https://github.com/GuardianAudits/GMXV1Updates

Table of Contents

Project Overview

Project Summary

Project Name	GMX
Language	Solidity
Codebase	GMX V1: https://github.com/gmx-io/gmx-contracts GMX V2: https://github.com/gmx-io/gmx-synthetics
Commit(s)	GMX V1 (initial): af8d73de2c05813f8f749ea41224d40cb6e94496 (final): c3bf7af2d37cb037c93270d26a45ef8b838fbf42 GMX V2 (initial): 18be26df12b64e67b15d938f2c0a327d392bc2c6 (final): beafb182044db94ba022bbd5605be33511905104

Audit Summary

Delivery Date	December 17, 2023
Audit Methodology	Static Analysis, Manual Review, Test Suite, Contract Fuzzing

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
● Critical	0	0	0	0	0	0
● High	1	0	0	1	0	0
● Medium	3	0	0	1	0	2
● Low	5	0	0	2	0	3

Audit Scope & Methodology

Vulnerability Classifications

Vulnerability Level	Classification
🔴 Critical	Easily exploitable by anyone, causing loss/manipulation of assets or data.
🟠 High	Arduously exploitable by a subset of addresses, causing loss/manipulation of assets or data.
🟡 Medium	Inherent risk of future exploits that may or may not impact the smart contract execution.
🟢 Low	Minor deviation from best practices.

Methodology

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross-referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.
- Comprehensive written tests as a part of a code coverage testing suite.
- Contract fuzzing for increased attack resilience.

Findings & Resolutions

ID	Title	Category	Severity	Status
RROU-1	Burning bnGMX Could Be Avoided	Logical Error	● High	Acknowledged
GLOBAL-1	Payload Attack Enables Griefing Of Keepers	Griefing	● Medium	Resolved
GLOBAL-2	Incorrect block.timestamp Used	Logical Error	● Medium	Resolved
RROU-2	Infinite Voting Power	Logical Error	● Medium	Acknowledged
RROU-3	Unnecessary Vote Syncing In _compound	Optimization	● Low	Acknowledged
RROU-4	Early Return Misses Voting Sync	Logical Error	● Low	Resolved
RROU-5	Unnecessary Voting Sync In _stakeGmx	Optimization	● Low	Resolved
RROU-6	inStrictTransferMode Does Not Ensure Successful acceptTransfer	Logic Error	● Low	Acknowledged
DEPLOY-1	Miniscule proposalThreshold Configured	Logical Error	● Low	Resolved

RROU-1 | Burning bnGMX Could Be Avoided

Category	Severity	Location	Status
Logical Error	● High	RewardRouterV2.sol: 505	Acknowledged

Description PoC

When unstaking GMX and `_shouldReduceBnGmx = true`, a proportionate amount of the user's bnGMX wallet balance is burnt.

A user can avoid the loss of their bnGMX (any amount beyond what is claimed in `_unstakeGmx`) by transferring the token to another account they control.

After unstaking, the sent bnGMX could be returned. Consequently, a user will have more multiplier points than intended, which will lead to more voting power.

Recommendation

Consider restricting the transfer of bnGMX for non-handler accounts. Furthermore, clearly document the intended behavior regarding bnGMX slashing upon unstaking GMX.

Resolution

GMX Team: bnGMX.inPrivateTransferMode is not currently set to true, agree that it should be set to true.

GLOBAL-1 | Payload Attack Enables Griefing Of Keepers

Category	Severity	Location	Status
Griefing	● Medium	Global	Resolved

Description

The `_transferOutETHWithGasLimitFallbackToWeth` and `_transferOutETH` functions in GMX V1 are used to transfer ether.

Neither of these functions utilizes assembly to avoid copying return data into memory.

This allows any contract interacting with GMX V1 to maliciously return a large amount of data, which will end up costing more gas than anticipated and could force the keepers to run a deficit.

Recommendation

Use a low-level call to avoid loading the return data into memory:
`assembly { success := call(gasLimit, receiver, amount, 0, 0, 0, 0) }`.

Resolution

GMX Team: The resolution was implemented.

GLOBAL-2 | Incorrect block.timestamp Used

Category	Severity	Location	Status
Logical Error	● Medium	Global	Resolved

Description

In the GovToken and ProtocolGovernance contracts, the CLOCK_MODE function validates that the clock function returns the block.timestamp, however it should be validated against the Chain.currentTimestamp to avoid failures due to L2 timestamp drift.

Recommendation

Validate the result of the clock function against the Chain.currentTimestamp().

Resolution

GMX Team: The resolution was implemented.

RROU-2 | Infinite Voting Power

Category	Severity	Location	Status
Logical Error	● Medium	RewardRouterV2.sol: 537	Acknowledged

Description PoC

When syncing an account's voting power, the user's staked amounts are compared against their current governance token wallet holdings. If the governance holdings are less, the appropriate amount is minted.

This can be exploited if a user transfers their governance tokens to another account, triggers a sync, and then those governance tokens are minted again. This way, a user can generate infinite votes and move forward malicious proposals.

Recommendation

Ensure that the governance token used cannot be freely transferred.

Resolution

GMX Team: GovToken.sol would be used, only contracts with the role GOV_TOKEN_CONTROLLER would be able to make transfers.

RROU-3 | Unnecessary Vote Syncing In _compound

Category	Severity	Location	Status
Optimization	● Low	RewardRouterV2.sol: 343	Acknowledged

Description

In the acceptTransfer function, the _compound invocation will always trigger a syncing of the voting power for the _sender. However, the voting is once again synced after stake balances are transferred at the end of the acceptTransfer function.

Recommendation

Consider adding a boolean parameter to the _compound function to specify whether the voting power should be synced. In the case of an account transfer this boolean would be false to avoid unnecessary syncs to save gas. In all other cases the boolean value would be true.

Resolution

GMX Team: Acknowledged.

RROU-4 | Early Return Misses Voting Sync

Category	Severity	Location	Status
Logical Error	● Low	RewardRouterV2.sol: 481	Resolved

Description

In the _stakeBnGmx function the _syncVotingPower invocation will be missed if the currentBnGmxAmount is greater than the maxAllowedBnGmxAmount.

There is no net effect as the _syncVotingPower function will always be called for the _account later for every instance where the _stakeBnGmx function is used.

However, this poses a risk if the _stakeBnGmx function were to be used without syncing the voting power of the _account afterwards.

Recommendation

Consider removing the _syncVotingPower invocation from the _stakeBnGmx function as it is redundant for all cases where the _stakeBnGmx function is currently used.

Otherwise be sure to sync the voting power in the case where the currentBnGmxAmount is greater than the maxAllowedBnGmxAmount and the _stakeBnGmx function early returns.

Resolution

GMX Team: The recommendation was implemented.

RROU-5 | Unnecessary Voting Sync In _stakeGmx

Category	Severity	Location	Status
Optimization	● Low	RewardRouterV2.sol: 446	Resolved

Description

In the _stakeGmx function it is unnecessary to sync the voting power for the funding account as the funding account will provide unstaked tokens, while the voting power relies on staked tokens.

Therefore the voting power of the funding account cannot be affected by the _stakeGmx function.

Additionally, in the only case where the _stakeGmx function is used with a _fundingAccount that is different from the account is in the acceptTransfer function where the voting power is redundantly synced for the _sender at the end.

Recommendation

Remove the syncing logic for the _fundingAccount in the _stakeGmx function as it is unnecessary.

Resolution

GMX Team: The recommendation was implemented.

RROU-6 | inStrictTransferMode Does Not Ensure Successful acceptTransfer

Category	Severity	Location	Status
Logic Error	● Low	RewardRouterV2.sol: 325	Acknowledged

Description

Calling the function signalTransfer requires that msg.sender has given the _receiver allowance equivalent to or more than the balance of msg.sender when inStrictTransferMode is set to true.

The original msg.sender can decrease the approval prior to the receiver accepting the transfer, causing the function acceptTransfer to revert in the case where the transfer is not being performed by a handler.

When the transfer is performed by the handler, transferFrom does not check the allowance on the BaseToken.

Recommendation

Consider removing the inStrictTransferMode check since the transfers are being done by handlers, or clearly document its intended behavior.

Resolution

GMX Team: The reason for inStrictTransferMode is mainly to prevent phishing, we have observed phishing scams that utilize signalTransfer, because no approval is needed, there is less warning / less alerting of the user from their wallet

DEPLOY-1 | Miniscule proposalThreshold Configured

Category	Severity	Location	Status
Logical Error	● Low	deployProtocolGovernor.ts: 15	Resolved

Description

In the deploy file for the ProtocolGovernor contract, 30_000 is used as a proposalThreshold. However, this is a minuscule amount of governance tokens, as the governance token is configured with 18 decimals.

Recommendation

Be sure to configure a reasonable minimum threshold with 18 decimals for proposals in production as the proposalThreshold is in a governance token amount, which has 18 decimals of precision.

Resolution

GMX Team: The recommendation was implemented.

Disclaimer

About Guardian Audits

Founded in 2022 by DeFi experts, Guardian Audits is a leading audit firm in the DeFi smart contract space. With every audit report, Guardian Audits upholds best-in-class security while achieving our mission to relentlessly secure DeFi.

To learn more, visit https://guardianaudits.com

To view our audit portfolio, visit https://github.com/guardianaudits

To book an audit, message https://t.me/guardianaudits