GUARDIAN AUDITS

SMART CONTRACT SECURITY AUDIT OF

GMX

Summary

Audit Firm Guardian

Prepared By Owen Thurm, Daniel Gelfand

Client Firm GMX

Final Report Date September 1, 2023

Audit Summary

GMX engaged Guardian to review the security of its real-time feed integration. From the 21st of August to the 1st of September, a team of 2 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

Table of Contents

Project Overview

Project Summary

Project Name	GMX
Language	Solidity
Codebase	https://github.com/gmx-io/gmx-synthetics
Commit(s)	c7af4b0c877bc4cb4f82ab040852111e9153605c

Audit Summary

Delivery Date	September 1, 2023
Audit Methodology	Static Analysis, Manual Review, Written Tests, Contract Fuzzing

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
High	0	0	0	0	0	0
Medium	3	3	0	0	0	0
Low	11	11	0	0	0	0

Audit Scope & Methodology

ID	File	SHA-1 Checksum(s)
CON	Config.sol	c8ecffd6defb76e89cd43a250d4697528926a871
TIME	Timelock.sol	5ef65a907382bd4a97670afbffdb31b917673f41
KEY	Keys.sol	ec5e13a8e8a84915b97ec1688c902bb4c80d417e
EDPU	ExecuteDepositUtils.sol	863ccc9b96cf533e257198088e3349e9df0fb1c0
ERR	Errors.sol	f6998a7e61b8843dbc9c9d0e9817aed2c4d90b16
ADL	AdlHandler.sol	997a862d7bb0a7aff77f8370f53e7cb1f20a069b
BOH	BaseOrderHandler.sol	1c29e866c18a7a6be3b27adb3f9cbaa1a86433ef
DEPH	DepositHandler.sol	b27d88814cc1d9c8bfc46eb6eb7c6213eeb581cf
WTDH	WithdrawalHandler.sol	e19c98f0036aa26e66d5e3197d36cdfd11b7f195
GSU	GasUtils.sol	55f63f3c8bcfdb5f20e156febd47cf3614a31ee2
MKTU	MarketUtils.sol	47deb4beb9a9a8d12b39dd62af6eabc00871c8d4
OCLU	OracleUtils.sol	3e25aaaa6865fe50a834ce875e371fdb5db97667
OCL	Oracle.sol	8c3bd8839197ea77e53a14929e3410c7cac0d9d6
ORDU	OrderUtils.sol	6a3021b06d65c790d23c7fda46db634888dcf045
DPCU	DecreasePositionCollateralUtils.sol	d8956a4b540e3459f75e4d16b1d8fe4b10d2960c
DPU	DecreasePositionUtils.sol	7f4fa2ff2bc69cf3c44dfec25a8891a513482064
SPU	SwapPricingUtils.sol	fd4d7efe4829aaf38878959756c7efcfdfc796d2
SWPU	SwapUtils.sol	c3bbd2100a3eac50bac5a1f0a0508c8489cc7601
EWDU	ExecuteWithdrawalUtils.sol	279df4e265a9da6238f7836e1349e520bc8c3794
WTDU	WithdrawalUtils.sol	e3c738577befac0f8f3bc1d62ee50efdad62285b

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
OCL-1	DoS Due To Crossed Markets	DoS	Medium	Unresolved
OCL-2	Verifier Configuration Risk-Free Trade	Protocol Manipulation	Medium	Unresolved
EDPU-1	Positive Impact Deposit Not Validated	Validation	Medium	Unresolved
OCLU-1	Unsorted Block Numbers	Logical Error	Low	Unresolved
OCLU-2	Typo	Typo	Low	Unresolved
OCL-3	Inaccurate Revert Data	Typo	Low	Unresolved
TIME-1	Inconsistent Realtime Feed Action Key	Typo	Low	Unresolved
OCL-4	Outdated Comment	Documentation	Low	Unresolved
OCL-5	Prices May Be Older Than The Allowed Age	Validation	Low	Unresolved
OCLU-3	Outdated NatSpec	Documentation	Low	Unresolved
OCL-6	Hardcoded VerifierProxy	Configuration	Low	Unresolved
OCL-7	Lack Of Realtime Feed Tokens Optimization	Optimization	Low	Unresolved
EDWU-1	Inconsistent Pool Value For Withdrawals	Events	Low	Unresolved
OCL-8	Bid/Ask Manipulation	Protocol Manipulation	Low	Unresolved

OCL-1 | DoS Due To Crossed Markets

Category	Severity	Location	Status
DoS	🟠 Medium	Oracle.sol: 561-563	Unresolved

Description

GMX reverts when the bid price is greater than the ask price, otherwise known as a crossed market. Crossed markets can typically happen in times of volatility, or when pricing is based on multiple venues/data providers. For example, if the highest bid is from Coinbase but the lowest ask is from Binance, the chance of a reported bid larger than the ask increases.

```
if (report.bid > report.ask) {
    revert Errors.InvalidRealtimeBidAsk(token, report.bid, report.ask);
}
```

Internal feeds cannot be used to regain protocol functionality because internal feeds cannot be enabled when a realtime feed is enabled for a particular token.

Recommendation

Verify whether the realtime feeds report crossed markets, if so consider an alternative to the strict validation or document and prepare for the risk of DoS.

Resolution

OCL-2 | Verifier Configuration Risk-Free Trade

Category	Severity	Location	Status
Protocol Manipulation	🟠 Medium	Oracle.sol: 549	Unresolved

Description

The configuration of the Chainlink `VerifierProxy` and `Verifier` contracts pose a non-trivial threat to the GMX V2 system.

A malicious user may observe any of the following scenarios and leverage them to execute a risk free trade on the platform:

- A certain feed used by GMX V2 is deactivated with `isDeactivated == true`
- A verifier has been unset with the `unsetVerifier` function for a particular `configDigest` used by GMX V2
- A particular config that corresponds to the `configDigest` used by GMX is not active with `isActive == false`

In any of the above scenarios a malicious actor is able to submit a market order which is un-executable during the period where the `feedId` or `configDigest` is deactivated/misconfigured. If market prices move against the trader during this time, the trader can simply cancel their order.

GMX V2 offers no alternative means of execution for these orders until the `feedId` or `configDigest` is reactivated as they cannot be executed with the regular oracle system and changing the oracle configuration for the tokens used would take days using the `Timelock` contract.

Recommendation

Consider implementing an alternative pathway to provide prices for orders in the event that a `feedId` or `configDigest` is deactivated or misconfigured. Otherwise consider disallowing the creation of orders that rely on certain `feedId`'s or `configDigest`'s that are currently deactivated.

Resolution

EDPU-1 | Positive Impact Deposit Not Validated

Category	Severity	Location	Status
Validation	🟠 Medium	ExecuteDepositUtils.sol: 409	Unresolved

Description

In the `_executeDeposit` function the pool amount is incremented by the positive price impact amount in the `_params.tokenOut`.

However only the `_params.tokenIn` is validated against the `validatePoolAmountForDeposit` validation. This could lead to the `tokenOut` balance exceeding the desired cap during deposits.

Recommendation

Validate that the increased `tokenOut` amount is also within the deposit cap with `validatePoolAmountForDeposit`.

Resolution

OCLU-1 | Unsorted Block Numbers

Category	Severity	Location	Status
Logical Error	🔶 Low	OracleUtils.sol: 210	Unresolved

Description

The `getUncompactedOracleBlockNumbers` function appends the min and max block numbers from realtime feeds to the end of the arrays. As a result, it is possible for the `minBlockNumbers` and `maxBlockNumbers` to be unsorted. This may cause ADL execution to be less predictable since the first `minBlockNumber` is selected to be the `updatedAtBlock` for the order:

```
cache.key = AdlUtils.createAdlOrder(
      AdlUtils.CreateAdlOrderParams(
            dataStore,
            . . .
            cache.minOracleBlockNumbers[0] <---
      )
);
```

For example:

ADL does not execute when the `cache.minOracleBlockNumbers` are [x, x+1] since `OracleUtils.validateBlockNumberWithinRange` prevent executions for the `updatedAtBlock` of x.

Now with the realtime feed block numbers appended, the `cache.minOracleBlockNumbers` can be unsorted as [x+1, x] with stale pricing used at block x and the ADL order will execute.

Recommendation

Consider having the block numbers sorted and/or document this potential behavior with the realtime reports.

Resolution

OCLU-2 | Typo

Category	Severity	Location	Status
Typo	🟠 Low	OracleUtils.sol: 85	Unresolved

Description

The comment “the highest price the a buyer will pay” should read “the highest price that a buyer will pay”.

Recommendation

Implement the above recommended changes.

Resolution

OCL-3 | Inaccurate Revert Data

Category	Severity	Location	Status
Typo	🟠 Low	Oracle.sol: 396	Unresolved

Description

The `Errors.InvalidBlockNumber` revert supplies the `minOracleBlockNumber` however it is the `maxOracleBlockNumber` which failed the block number validation.

Recommendation

Revert with the `maxOracleBlockNumber` as the invalid block number.

Resolution

TIME-1 | Inconsistent Realtime Feed Action Key

Category	Severity	Location	Status
Typo	🔶 Low	Timelock.sol: 479	Unresolved

Description

In the `_setRealtimeFeedActionKey` function the “setPriceFeed” string is used to create the realtime feed action key. However following from the patterns of the other action keys, the realtime feed action key ought to use the string “setRealtimeFeed” to match the `actionLabel`.

Recommendation

Use the “setRealtimeFeed” string to construct the realtime feed action key in the `_setRealtimeFeedActionKey` function.

Resolution

OCL-4 | Outdated Comment

Category	Severity	Location	Status
Documentation	🟠 Low	Oracle.sol: 334	Unresolved

Description

The comment above the `_setPrices` function is outdated, it references initializing a `SetPricesCache` which does not happen immediately in the `_setPrices` function. Additionally there is no referenced `signers` param.

Recommendation

Update the documentation for the `_setPrices` function.

Resolution

OCL-5 | Prices May Be Older Than The Allowed Age

Category	Severity	Location	Status
Validation	🔶 Low	Oracle.sol: 572	Unresolved

Description

The `maxPriceAge` validation is performed on the `currentBlockTimestamp` which is based on the upper bound block number. Therefore bid and ask values can technically come from before the `maxPriceAge` window, though perhaps a trivial amount of time.

Recommendation

Be aware that the `maxPriceAge` can be slightly exceeded and configure the `maxPriceAge` as such.

Resolution

OCLU-3 | Outdated NatSpec

Category	Severity	Location	Status
Documentation	🔶 Low	OracleUtils.sol: 206	Unresolved

Description

The NatSpec for the `getUncompactedOracleBlockNumbers` function is outdated, it includes the wrong name for the `compactedOracleBlockNumbersLength` parameter and lacks reference to the reports and `oracleBlockNumberType` parameters.

Recommendation

Update the NatSpec for the `getUncompactedOracleBlockNumbers` function.

Resolution

OCL-6 | Hardcoded VerifierProxy

Category	Severity	Location	Status
Configuration	🔶 Low	Oracle.sol: 86	Unresolved

Description

In the event that a new `VerifierProxy` is deployed by Chainlink the Oracle contract would have to be redeployed as the `realtimeFeedVerifier` variable is `immutable`.

However it may be more wieldy to allow the `realtimeFeedVerifier` implementation to be configurable in the event that a new proxy should be used.

Recommendation

Consider allowing the `realtimeFeedVerifier` implementation to be configurable in the `dataStore` rather than `immutable`.

Resolution

OCL-7 | Lack Of Realtime Feed Tokens Optimization

Category	Severity	Location	Status
Optimization	🔶 Low	Oracle.sol: 525	Unresolved

Description

In the event that there are no `realtimeFeedTokens`, the `_validateRealtimeFeeds` function logic can be shortcut to avoid expending gas on extra opcodes.

Recommendation

Consider implementing a `realtimeFeedTokens.length == 0` early return statement at the beginning of the `_validateRealtimeFeeds` function.

Resolution

EWDU-1 | Inconsistent Pool Value For Withdrawals

Category	Severity	Location	Status
Events	🔶 Low	ExecuteWithdrawalUtils.sol: 321-329	Unresolved

Description

In the `_getOutputAmounts` function the pool value information emitted with the `MarketPoolValueUpdated` event is obtained with the following call:

```
MarketPoolValueInfo.Props memory poolValueInfo = MarketUtils.getPoolValueInfo(
            params.dataStore,
            . . .
            Keys.MAX_PNL_FACTOR_FOR_WITHDRAWALS,
            false
        );
```

This differs from the call to `MarketUtils.getPoolValueInfo` in `_executeWithdrawal` since the `pnlFactorType` is set to `Keys.MAX_PNL_FACTOR_FOR_DEPOSITS` and the value is maximized:

```
MarketPoolValueInfo.Props memory poolValueInfo = MarketUtils.getPoolValueInfo(
            params.dataStore,
            . . .
            Keys.MAX_PNL_FACTOR_FOR_DEPOSITS, <---
            true <---
        );
```

Recommendation

Consider whether these parameters are desired for the data emitted in the `MarketPoolValueUpdated` event. If they are not, modify the `pnlFactorType` to `Keys.MAX_PNL_FACTOR_FOR_WITHDRAWALS` and set maximize to `false`.

Resolution

OCL-8 | Bid/Ask Manipulation

Category	Severity	Location	Status
Protocol Manipulation	🟠 Low	Oracle.sol	Unresolved

Description

Execution prices with realtime feeds are based on bid/ask spreads from a collection of reference exchanges, however, they do not consider the liquidity at any given bid or ask price. Therefore, a malicious actor may manipulate the lowest ask or increase the highest bid with a relatively small liquidity on a reference exchange and trade based on that synthesized lowest ask or highest bid with a significant size on GMX V2.

Any potential manipulation is however limited in magnitude to the spread between the bids and asks.

Recommendation

Ensure that there is a sufficiently high minimum liquidity barrier to consider a bid or ask and that fees and price impact on the GMX V2 platform can effectively counteract the profitability of any such manipulation.

Resolution

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.

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