// HALBORN

MatterLabs - Verifier

Smart Contract Security Assessment

Prepared by: **Halborn**

Date of Engagement: **July 12th, 2023 - July 20th, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	07/19/2023	Omar Alshaeb
0.2	Document Updates	07/20/2023	Omar Alshaeb
0.3	Draft Review	07/21/2023	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Gokberk Gulgun	Halborn	Gokberk.Gulgun@halborn.com
Omar Alshaeb	Halborn	Omar.Alshaeb@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

This assessment was entirely focused on the new version of zkSync verifier which is a modified version of the Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge (PLONK) to optimize the proof system for zkSync Era circuits.

MatterLabs engaged Halborn to conduct a security assessment on their verifier smart contract beginning on July 12th, 2023 and ending on July 20th, 2023. The security assessment was scoped to the smart contract provided to the Halborn team.

1.2 ASSESSMENT SUMMARY

The team at Halborn was provided one week for the engagement and assigned a full-time security engineer to assessment the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this assessment is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues within the smart contract

In summary, Halborn did not identify any security risks within the verifier smart contract.

1.3 SCOPE

1. IN-SCOPE:

The security assessment was scoped to the following smart contract:

- ethereum/contracts/verifier/Verifier.sol

Commit ID: f783f571e16a1b1adddb13db45db741f83b94812

1.4 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this assessment. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the bridge code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the assessment:

- Research into architecture and purpose.
- Smart contract manual code review and walkthrough.
- Graphing out functionality and contract logic/connectivity/functions. (solgraph)
- Manual assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes.
- Manual testing by custom scripts.
- Scanning of solidity files for vulnerabilities, security hotspots or bugs. (MythX)
- Static Analysis of security for scoped contract, and imported functions. (Slither)
- Testnet deployment. (Foundry)

2. RISK METHODOLOGY

Every vulnerability and issue observed by Halborn is ranked based on **two sets** of **Metrics** and a **Severity Coefficient**. This system is inspired by the industry standard Common Vulnerability Scoring System.

The two **Metric sets** are: **Exploitability** and **Impact**. **Exploitability** captures the ease and technical means by which vulnerabilities can be exploited and **Impact** describes the consequences of a successful exploit.

The **Severity Coefficients** is designed to further refine the accuracy of the ranking with two factors: **Reversibility** and **Scope**. These capture the impact of the vulnerability on the environment as well as the number of users and smart contracts affected.

The final score is a value between 0-10 rounded up to 1 decimal place and 10 corresponding to the highest security risk. This provides an objective and accurate rating of the severity of security vulnerabilities in smart contracts.

The system is designed to assist in identifying and prioritizing vulnerabilities based on their level of risk to address the most critical issues in a timely manner.

2.1 EXPLOITABILITY

Attack Origin (AO):

Captures whether the attack requires compromising a specific account.

Attack Cost (AC):

Captures the cost of exploiting the vulnerability incurred by the attacker relative to sending a single transaction on the relevant blockchain. Includes but is not limited to financial and computational cost.

Attack Complexity (AX):

Describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability. Includes but is not limited to macro situation, available third-party liquidity and regulatory challenges.

Metrics:

Exploitability Metric (m_E)	Metric Value	Numerical Value
Attack Origin (AO)	Arbitrary (AO:A) Specific (AO:S)	1 0.2
Attack Cost (AC)	Low (AC:L) Medium (AC:M) High (AC:H)	1 0.67 0.33
Attack Complexity (AX)	Low (AX:L) Medium (AX:M) High (AX:H)	1 0.67 0.33

Exploitability E is calculated using the following formula:

$$E = \prod m_e$$

2.2 IMPACT

Confidentiality (C):

Measures the impact to the confidentiality of the information resources managed by the contract due to a successfully exploited vulnerability. Confidentiality refers to limiting access to authorized users only.

Integrity (I):

Measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and veracity of data stored and/or processed on-chain. Integrity impact directly affecting Deposit or Yield records is excluded.

Availability (A):

Measures the impact to the availability of the impacted component resulting from a successfully exploited vulnerability. This metric refers to smart contract features and functionality, not state. Availability impact directly affecting Deposit or Yield is excluded.

Deposit (D):

Measures the impact to the deposits made to the contract by either users or owners.

Yield (Y):

Measures the impact to the yield generated by the contract for either users or owners.

Metrics:

Impact Metric (m_I)	Metric Value	Numerical Value
Confidentiality (C)	None (I:N)	0
	Low (I:L)	0.25
	Medium (I:M)	0.5
	High (I:H)	0.75
	Critical (I:C)	1
Integrity (I)	None (I:N)	0
	Low (I:L)	0.25
	Medium (I:M)	0.5
	High (I:H)	0.75
	Critical (I:C)	1
Availability (A)	None (A:N)	0
	Low (A:L)	0.25
	Medium (A:M)	0.5
	High (A:H)	0.75
	Critical	1
Deposit (D)	None (D:N)	0
	Low (D:L)	0.25
	Medium (D:M)	0.5
	High (D:H)	0.75
	Critical (D:C)	1
Yield (Y)	None (Y:N)	0
	Low (Y:L)	0.25
	Medium: (Y:M)	0.5
	High: (Y:H)	0.75
	Critical (Y:H)	1

Impact I is calculated using the following formula:

$$I = max(m_I) + \frac{\sum m_I - max(m_I)}{4}$$

2.3 SEVERITY COEFFICIENT

Reversibility (R):

Describes the share of the exploited vulnerability effects that can be reversed. For upgradeable contracts, assume the contract private key is available.

Scope (S):

Captures whether a vulnerability in one vulnerable contract impacts resources in other contracts.

Coefficient (C)	Coefficient Value	Numerical Value
Reversibility (r)	None (R:N) Partial (R:P) Full (R:F)	1 0.5 0.25
Scope (s)	Changed (S:C) Unchanged (S:U)	1.25 1

Severity Coefficient C is obtained by the following product:

$$C = rs$$

The Vulnerability Severity Score S is obtained by:

$$S = min(10, EIC * 10)$$

The score is rounded up to 1 decimal places.

Severity	Score Value Range
Critical	9 - 10
High	7 - 8.9
Medium	4.5 - 6.9
Low	2 - 4.4
Informational	0 - 1.9

3. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	0	0

MANUAL TESTING

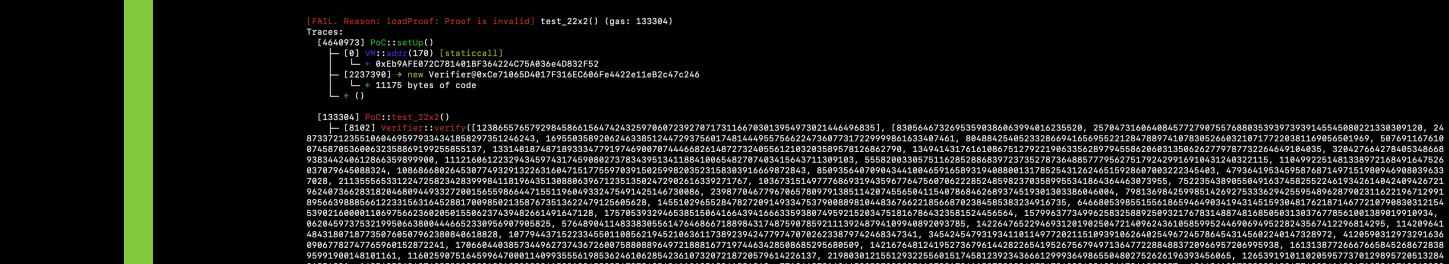
The main goal of the manual testing performed during this assessment was to test that the verifier is properly working to verify the zk proofs generated by the zkSync Era circuits, focusing on the following points/scenarios:

Test	Result
Check that using any valid proof, the verifier is able to properly verify it and returns a true as a result	Pass

MANUAL TESTING

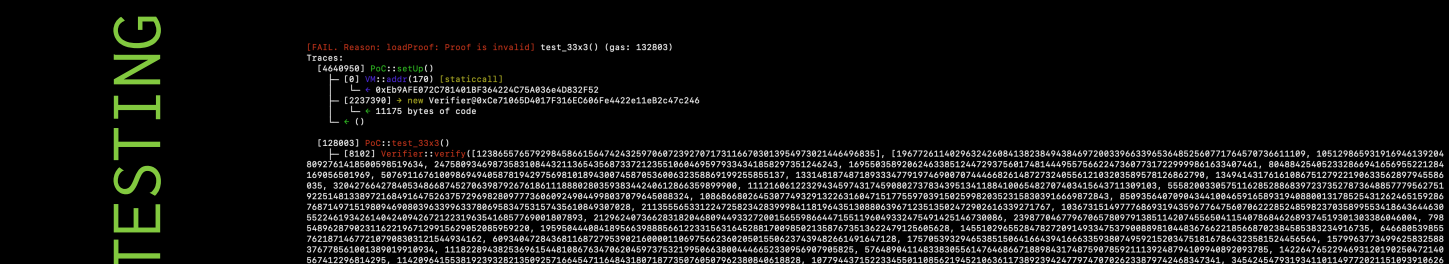
MANUAL TESTING

Test	Result
Check that verifier reverts with proof is invalid if a maliciously forged serialized proof is sent	Pass



Test	Result
Check that verifier reverts with proof is invalid if less than 44 words for serialized proof is sent	Pass

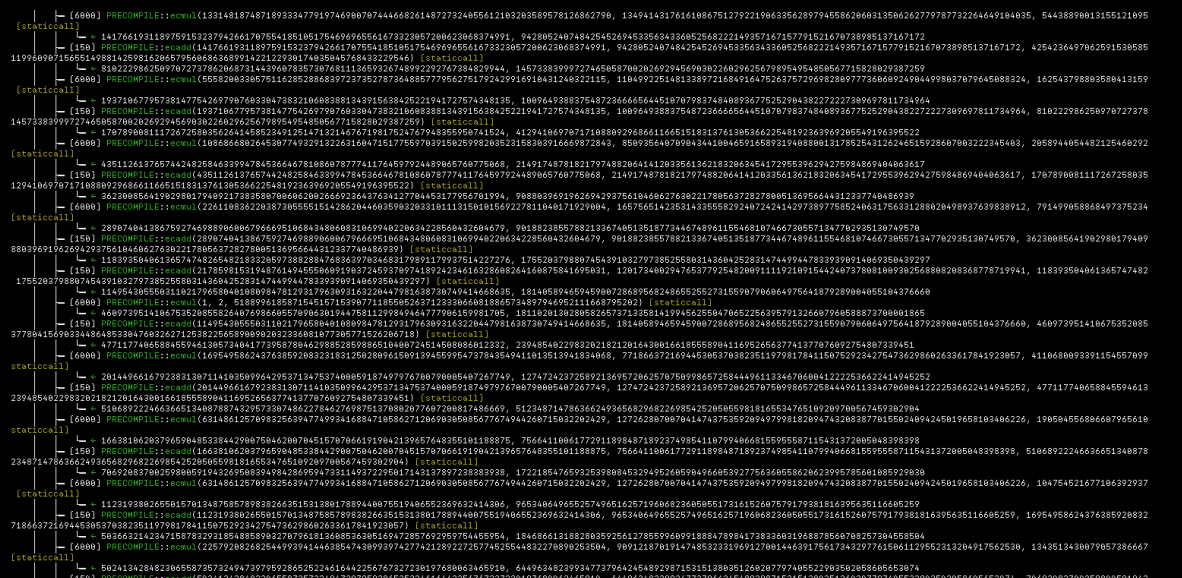
Test	Result
Check that verifier reverts with proof is invalid if less than 4 words for recursive aggregation input is sent	Pass



Test	Result
Check that verifier returns a true for a public input with dirty bits over Fr mask	Pass

MANUAL TESTING

Test	Result
Check that the verifier returns a true having elliptic curve points over modulo	Pass



MANUAL TESTING

MANUAL TESTING

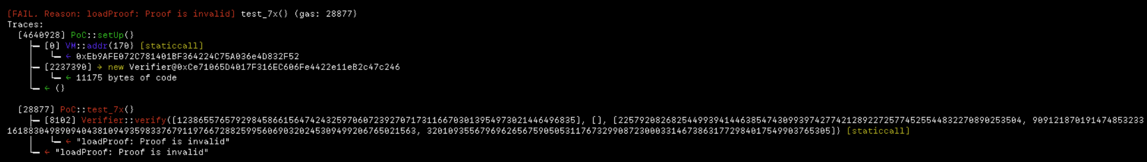
Test	Result
Check that the verifier returns a true, having Fr over modulo	Pass

Test	Result
Check that verifier reverts with proof is invalid if more than 1 public inputs is sent	Pass

Test	Result
Check that verifier reverts with proof is invalid if empty public inputs is sent	Pass

Test	Result
Check that verifier reverts with proof is invalid if more than 44 words for serialized proof is sent	Pass

Test	Result
Check that verifier reverts with proof is invalid if empty serialized proof is sent	Pass



Test	Result
Check that verifier reverts with proof is invalid if more than 4 words for recursive aggregation input is sent	Pass

Test	Result
Check that verifier reverts with proof is invalid if empty recursive aggregation input is sent	Pass

MANUAL TESTING

Test	Result
Check that verifier reverts with proof is invalid if elliptic curve point at infinity is sent within the serialized proof	Pass

Test	Result
Check that the verifier reverts with invalid quotient evaluation if an invalid public input is used	Pass

MANUAL TESTING

Test	Result
Check that the verifier reverts with pairing failure if an invalid recursive aggregative input is used	Pass

AUTOMATED TESTING

5.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the scoped contract. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified the contract in the repository and was able to compile it correctly into their ABI and binary formats, Slither was run on the verifier contract. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.

Slither results:

ethereum/contracts/verifier/Verifier.sol

```
Verifier.verificationKeyHash() (src/verifier/Verifier.sol#253-263) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#256-262)
Verifier._loadVerificationKey() (src/verifier/Verifier.sol#276-324) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#277-323)
Verifier.verify(uint256[],uint256[],uint256[]) (src/verifier/Verifier.sol#329-1671) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#339-1670)
Verifier.verify.asm_0.revertWithMessage() (src/verifier/Verifier.sol#347-358) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#347-358)
Verifier.verify.asm_0.modexp() (src/verifier/Verifier.sol#361-372) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#361-372)
Verifier.verify.asm_0.pointMulIntoDest() (src/verifier/Verifier.sol#375-382) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#375-382)
Verifier.verify.asm_0.pointAddIntoDest() (src/verifier/Verifier.sol#385-393) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#385-393)
Verifier.verify.asm_0.pointSubAssign() (src/verifier/Verifier.sol#396-404) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#396-404)
Verifier.verify.asm_0.pointAddAssign() (src/verifier/Verifier.sol#407-415) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#407-415)
Verifier.verify.asm_0.pointMulAndAddIntoDest() (src/verifier/Verifier.sol#418-431) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#418-431)
Verifier.verify.asm_0.pointNegate() (src/verifier/Verifier.sol#434-445) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#434-445)
Verifier.verify.asm_0.updateTranscript() (src/verifier/Verifier.sol#452-460) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#452-460)
Verifier.verify.asm_0.getTranscriptChallenge() (src/verifier/Verifier.sol#463-467) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#463-467)
Verifier.verify.asm_0.loadProof() (src/verifier/Verifier.sol#499-682) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#499-682)
Verifier.verify.asm_0.initializeTranscript() (src/verifier/Verifier.sol#710-794) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#710-794)
Verifier.verify.asm_0.verifyQuotientEvaluation() (src/verifier/Verifier.sol#818-860) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#818-860)
Verifier.verify.asm_0.evaluateLagrangePolyOutOfDomain() (src/verifier/Verifier.sol#865-882) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#865-882)
Verifier.verify.asm_0.permutationQuotientContribution() (src/verifier/Verifier.sol#885-943) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#885-943)
Verifier.verify.asm_0.lookupQuotientContribution() (src/verifier/Verifier.sol#946-988) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#946-988)
Verifier.verify.asm_0.mainGateLinearisationContributionWithV() (src/verifier/Verifier.sol#991-1018) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#991-1018)
Verifier.verify.asm_0.addAssignRescueCustomGateLinearisationContributionWithV() (src/verifier/Verifier.sol#1021-1048) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#1021-1048)
Verifier.verify.asm_0.addAssignPermutationLinearisationContributionWithV() (src/verifier/Verifier.sol#1051-1152) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#1051-1152)
Verifier.verify.asm_0.addAssignLookupLinearisationContributionWithV() (src/verifier/Verifier.sol#1155-1227) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#1155-1227)
Verifier.verify.asm_0.prepareQueries() (src/verifier/Verifier.sol#1263-1339) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#1263-1339)
Verifier.verify.asm_0.finalPairing() (src/verifier/Verifier.sol#1591-1644) uses assembly
        - INLINE ASM (src/verifier/Verifier.sol#1591-1644)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Verifier.verify.asm_0.pointAddIntoDest() (src/verifier/Verifier.sol#385-393) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.13 (src/verifier/IVerifier.sol#3) allows old versions
Pragma version^0.8.13 (src/verifier/Verifier.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Parameter Verifier.verify.asm_0.revertWithMessage().len_verify_asm_0_revertWithMessage (src/verifier/Verifier.sol#347) is not in mixedCase
Parameter Verifier.verify.asm_0.revertWithMessage().reason_verify_asm_0_revertWithMessage (src/verifier/Verifier.sol#347) is not in mixedCase
Parameter Verifier.verify.asm_0.modexp().value_verify_asm_0_modexp (src/verifier/Verifier.sol#361) is not in mixedCase
Parameter Verifier.verify.asm_0.modexp().power_verify_asm_0_modexp (src/verifier/Verifier.sol#361) is not in mixedCase
Parameter Verifier.verify.asm_0.pointMulIntoDest().point_verify_asm_0_pointMulIntoDest (src/verifier/Verifier.sol#375) is not in mixedCase
Parameter Verifier.verify.asm_0.pointMulIntoDest().s_verify_asm_0_pointMulIntoDest (src/verifier/Verifier.sol#375) is not in mixedCase
Parameter Verifier.verify.asm_0.pointMulIntoDest().dest_verify_asm_0_pointMulIntoDest (src/verifier/Verifier.sol#375) is not in mixedCase
Parameter Verifier.verify.asm_0.pointAddIntoDest().p1_verify_asm_0_pointAddIntoDest (src/verifier/Verifier.sol#385) is not in mixedCase
Parameter Verifier.verify.asm_0.pointAddIntoDest().p2_verify_asm_0_pointAddIntoDest (src/verifier/Verifier.sol#385) is not in mixedCase
Parameter Verifier.verify.asm_0.pointAddIntoDest().dest_verify_asm_0_pointAddIntoDest (src/verifier/Verifier.sol#385) is not in mixedCase
Parameter Verifier.verify.asm_0.pointSubAssign().p1_verify_asm_0_pointSubAssign (src/verifier/Verifier.sol#396) is not in mixedCase
Parameter Verifier.verify.asm_0.pointSubAssign().p2_verify_asm_0_pointSubAssign (src/verifier/Verifier.sol#396) is not in mixedCase
Parameter Verifier.verify.asm_0.pointAddAssign().p1_verify_asm_0_pointAddAssign (src/verifier/Verifier.sol#407) is not in mixedCase
Parameter Verifier.verify.asm_0.pointAddAssign().p2_verify_asm_0_pointAddAssign (src/verifier/Verifier.sol#407) is not in mixedCase
```

```
Parameter Verifier.verify.asm_0.pointMulAndAddIntoDest().point_verify_asm_0_pointMulAndAddIntoDest (src/verifier/Verifier.sol#418) is not in mixedCase
Parameter Verifier.verify.asm_0.pointMulAndAddIntoDest().s_verify_asm_0_pointMulAndAddIntoDest (src/verifier/Verifier.sol#418) is not in mixedCase
Parameter Verifier.verify.asm_0.pointMulAndAddIntoDest().dest_verify_asm_0_pointMulAndAddIntoDest (src/verifier/Verifier.sol#418) is not in mixedCase
Parameter Verifier.verify.asm_0.pointNegate().point_verify_asm_0_pointNegate (src/verifier/Verifier.sol#434) is not in mixedCase
Parameter Verifier.verify.asm_0.updateTranscript().value_verify_asm_0_updateTranscript (src/verifier/Verifier.sol#452) is not in mixedCase
Parameter Verifier.verify.asm_0.getTranscriptChallenge().numberOfChallenge_verify_asm_0_getTranscriptChallenge (src/verifier/Verifier.sol#463) is not in mixedCase
Parameter Verifier.verify.asm_0.evaluateLagrangePolyOutOfDomain().polyNum_verify_asm_0_evaluateLagrangePolyOutOfDomain (src/verifier/Verifier.sol#865) is not in mixedCase
Parameter Verifier.verify.asm_0.evaluateLagrangePolyOutOfDomain().at_verify_asm_0_evaluateLagrangePolyOutOfDomain (src/verifier/Verifier.sol#865) is not in mixedCase
Parameter Verifier.verify.asm_0.mainGateLinearisationContributionWithV().dest_verify_asm_0_mainGateLinearisationContributionWithV (src/verifier/Verifier.sol#992) is not in mixedCase
Parameter Verifier.verify.asm_0.mainGateLinearisationContributionWithV().stateOpening0At2_verify_asm_0_mainGateLinearisationContributionWithV (src/verifier/Verifier.sol#993) is not in mixedCase
Parameter Verifier.verify.asm_0.mainGateLinearisationContributionWithV().stateOpening1At2_verify_asm_0_mainGateLinearisationContributionWithV (src/verifier/Verifier.sol#994) is not in mixedCase
Parameter Verifier.verify.asm_0.mainGateLinearisationContributionWithV().stateOpening2At2_verify_asm_0_mainGateLinearisationContributionWithV (src/verifier/Verifier.sol#995) is not in mixedCase
Parameter Verifier.verify.asm_0.mainGateLinearisationContributionWithV().stateOpening3At2_verify_asm_0_mainGateLinearisationContributionWithV (src/verifier/Verifier.sol#996) is not in mixedCase
Parameter Verifier.verify.asm_0.addAssignRescueCustomGateLinearisationContributionWithV().dest_verify_asm_0_addAssignRescueCustomGateLinearisationContributionWithV (src/verifier/Verifier.sol#1022) is not in mixedCa
Parameter Verifier.verify.asm_0.addAssignRescueCustomGateLinearisationContributionWithV().stateOpening0At2_verify_asm_0_addAssignRescueCustomGateLinearisationContributionWithV (src/verifier/Verifier.sol#1023) is no
Parameter Verifier.verify.asm_0.addAssignRescueCustomGateLinearisationContributionWithV().stateOpening1At2_verify_asm_0_addAssignRescueCustomGateLinearisationContributionWithV (src/verifier/Verifier.sol#1024) is no
Parameter Verifier.verify.asm_0.addAssignRescueCustomGateLinearisationContributionWithV().stateOpening2At2_verify_asm_0_addAssignRescueCustomGateLinearisationContributionWithV (src/verifier/Verifier.sol#1025) is no
Parameter Verifier.verify.asm_0.addAssignRescueCustomGateLinearisationContributionWithV().stateOpening3At2_verify_asm_0_addAssignRescueCustomGateLinearisationContributionWithV (src/verifier/Verifier.sol#1026) is no
Parameter Verifier.verify.asm_0.addAssignPermutationLinearisationContributionWithV().dest_verify_asm_0_addAssignPermutationLinearisationContributionWithV (src/verifier/Verifier.sol#1052) is not in mixedCase
Parameter Verifier.verify.asm_0.addAssignPermutationLinearisationContributionWithV().stateOpening0At2_verify_asm_0_addAssignPermutationLinearisationContributionWithV (src/verifier/Verifier.sol#1053) is not in mixed
Parameter Verifier.verify.asm_0.addAssignPermutationLinearisationContributionWithV().stateOpening1At2_verify_asm_0_addAssignPermutationLinearisationContributionWithV (src/verifier/Verifier.sol#1054) is not in mixed
Parameter Verifier.verify.asm_0.addAssignPermutationLinearisationContributionWithV().stateOpening2At2_verify_asm_0_addAssignPermutationLinearisationContributionWithV (src/verifier/Verifier.sol#1055) is not in mixed
Parameter Verifier.verify.asm_0.addAssignPermutationLinearisationContributionWithV().stateOpening3At2_verify_asm_0_addAssignPermutationLinearisationContributionWithV (src/verifier/Verifier.sol#1056) is not in mixed
Parameter Verifier.verify.asm_0.addAssignLookupLinearisationContributionWithV().dest_verify_asm_0_addAssignLookupLinearisationContributionWithV (src/verifier/Verifier.sol#1156) is not in mixedCase
Parameter Verifier.verify.asm_0.addAssignLookupLinearisationContributionWithV().stateOpening0At2_verify_asm_0_addAssignLookupLinearisationContributionWithV (src/verifier/Verifier.sol#1157) is not in mixedCase
Parameter Verifier.verify.asm_0.addAssignLookupLinearisationContributionWithV().stateOpening1At2_verify_asm_0_addAssignLookupLinearisationContributionWithV (src/verifier/Verifier.sol#1158) is not in mixedCase
Parameter Verifier.verify.asm_0.addAssignLookupLinearisationContributionWithV().stateOpening2At2_verify_asm_0_addAssignLookupLinearisationContributionWithV (src/verifier/Verifier.sol#1159) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable Verifier.AGGREGATED_AT_Z_OMEGA_X_SLOT (src/verifier/Verifier.sol#193) is too similar to Verifier.AGGREGATED_AT_Z_OMEGA_Y_SLOT (src/verifier/Verifier.sol#194)
Variable Verifier.AGGREGATED_AT_Z_X_SLOT (src/verifier/Verifier.sol#190) is too similar to Verifier.AGGREGATED_AT_Z_Y_SLOT (src/verifier/Verifier.sol#191)
Variable Verifier.NON_RESIDUES_0 (src/verifier/Verifier.sol#237) is too similar to Verifier.NON_RESIDUES_2 (src/verifier/Verifier.sol#239)
Variable Verifier.NON_RESIDUES_0 (src/verifier/Verifier.sol#237) is too similar to Verifier.NON_RESIDUES_1 (src/verifier/Verifier.sol#238)
Variable Verifier.NON_RESIDUES_1 (src/verifier/Verifier.sol#238) is too similar to Verifier.NON_RESIDUES_2 (src/verifier/Verifier.sol#239)
Variable Verifier.PAIRING_BUFFER_POINT_X_SLOT (src/verifier/Verifier.sol#203) is too similar to Verifier.PAIRING_BUFFER_POINT_Y_SLOT (src/verifier/Verifier.sol#204)
Variable Verifier.PAIRING_PAIR_WITH_GENERATOR_X_SLOT (src/verifier/Verifier.sol#206-207) is too similar to Verifier.PAIRING_PAIR_WITH_GENERATOR_Y_SLOT (src/verifier/Verifier.sol#208-209)
Variable Verifier.PAIRING_PAIR_WITH_X_X_SLOT (src/verifier/Verifier.sol#211) is too similar to Verifier.PAIRING_PAIR_WITH_X_Y_SLOT (src/verifier/Verifier.sol#212)
Variable Verifier.PROOF_COPY_PERMUTATION_GRAND_PRODUCT_X_SLOT (src/verifier/Verifier.sol#84) is too similar to Verifier.PROOF_COPY_PERMUTATION_GRAND_PRODUCT_Y_SLOT (src/verifier/Verifier.sol#85)
Variable Verifier.PROOF_COPY_PERMUTATION_POLYS_0_OPENING_AT_Z_SLOT (src/verifier/Verifier.sol#110) is too similar to Verifier.PROOF_COPY_PERMUTATION_POLYS_1_OPENING_AT_Z_SLOT (src/verifier/Verifier.sol#111)
Variable Verifier.PROOF_COPY_PERMUTATION_POLYS_0_OPENING_AT_Z_SLOT (src/verifier/Verifier.sol#110) is too similar to Verifier.PROOF_COPY_PERMUTATION_POLYS_2_OPENING_AT_Z_SLOT (src/verifier/Verifier.sol#112)
Variable Verifier.PROOF_COPY_PERMUTATION_POLYS_1_OPENING_AT_Z_SLOT (src/verifier/Verifier.sol#111) is too similar to Verifier.PROOF_COPY_PERMUTATION_POLYS_2_OPENING_AT_Z_SLOT (src/verifier/Verifier.sol#112)
Variable Verifier.PROOF_LOOKUP_GRAND_PRODUCT_X_SLOT (src/verifier/Verifier.sol#90) is too similar to Verifier.PROOF_LOOKUP_GRAND_PRODUCT_Y_SLOT (src/verifier/Verifier.sol#91)
Variable Verifier.PROOF_LOOKUP_S_POLY_X_SLOT (src/verifier/Verifier.sol#87) is too similar to Verifier.PROOF_LOOKUP_S_POLY_Y_SLOT (src/verifier/Verifier.sol#88)
Variable Verifier.PROOF_LOOKUP_S_POLY_OPENING_AT_Z_OMEGA_SLOT (src/verifier/Verifier.sol#115) is too similar to Verifier.PROOF_LOOKUP_T_POLY_OPENING_AT_Z_OMEGA_SLOT (src/verifier/Verifier.sol#118)
Variable Verifier.PROOF_OPENING_PROOF_AT_Z_OMEGA_X_SLOT (src/verifier/Verifier.sol#126) is too similar to Verifier.PROOF_OPENING_PROOF_AT_Z_OMEGA_Y_SLOT (src/verifier/Verifier.sol#127)
Variable Verifier.PROOF_OPENING_PROOF_AT_Z_X_SLOT (src/verifier/Verifier.sol#124) is too similar to Verifier.PROOF_OPENING_PROOF_AT_Z_Y_SLOT (src/verifier/Verifier.sol#125)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_0_X_SLOT (src/verifier/Verifier.sol#93) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_0_Y_SLOT (src/verifier/Verifier.sol#94)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_0_X_SLOT (src/verifier/Verifier.sol#93) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_1_X_SLOT (src/verifier/Verifier.sol#95)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_0_Y_SLOT (src/verifier/Verifier.sol#94) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_1_X_SLOT (src/verifier/Verifier.sol#95)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_0_X_SLOT (src/verifier/Verifier.sol#93) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_1_Y_SLOT (src/verifier/Verifier.sol#96)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_0_Y_SLOT (src/verifier/Verifier.sol#94) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_1_Y_SLOT (src/verifier/Verifier.sol#96)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_1_X_SLOT (src/verifier/Verifier.sol#95) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_1_Y_SLOT (src/verifier/Verifier.sol#96)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_1_Y_SLOT (src/verifier/Verifier.sol#96) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_2_X_SLOT (src/verifier/Verifier.sol#97)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_1_Y_SLOT (src/verifier/Verifier.sol#96) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_2_Y_SLOT (src/verifier/Verifier.sol#98)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_1_Y_SLOT (src/verifier/Verifier.sol#96) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_3_X_SLOT (src/verifier/Verifier.sol#99)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_2_X_SLOT (src/verifier/Verifier.sol#97) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_2_Y_SLOT (src/verifier/Verifier.sol#98)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_2_X_SLOT (src/verifier/Verifier.sol#97) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_3_X_SLOT (src/verifier/Verifier.sol#99)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_2_Y_SLOT (src/verifier/Verifier.sol#98) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_3_X_SLOT (src/verifier/Verifier.sol#99)
Variable Verifier.PROOF_QUOTIENT_POLY_PARTS_3_X_SLOT (src/verifier/Verifier.sol#99) is too similar to Verifier.PROOF_QUOTIENT_POLY_PARTS_3_Y_SLOT (src/verifier/Verifier.sol#100)
Variable Verifier.VK_LOOKUP_TABLE_0_Y_SLOT (src/verifier/Verifier.sol#58) is too similar to Verifier.VK_LOOKUP_TABLE_1_Y_SLOT (src/verifier/Verifier.sol#60)
Variable Verifier.VK_LOOKUP_TABLE_1_X_SLOT (src/verifier/Verifier.sol#59) is too similar to Verifier.VK_LOOKUP_TABLE_1_Y_SLOT (src/verifier/Verifier.sol#60)
Variable Verifier.VK_LOOKUP_TABLE_1_Y_SLOT (src/verifier/Verifier.sol#60) is too similar to Verifier.VK_LOOKUP_TABLE_3_Y_SLOT (src/verifier/Verifier.sol#64)
Variable Verifier.VK_LOOKUP_TABLE_0_X_SLOT (src/verifier/Verifier.sol#57) is too similar to Verifier.VK_LOOKUP_TABLE_2_X_SLOT (src/verifier/Verifier.sol#61)
Variable Verifier.VK_LOOKUP_TABLE_0_Y_SLOT (src/verifier/Verifier.sol#58) is too similar to Verifier.VK_LOOKUP_TABLE_2_X_SLOT (src/verifier/Verifier.sol#61)
Variable Verifier.VK_LOOKUP_TABLE_1_X_SLOT (src/verifier/Verifier.sol#59) is too similar to Verifier.VK_LOOKUP_TABLE_2_X_SLOT (src/verifier/Verifier.sol#61)
Variable Verifier.VK_LOOKUP_TABLE_1_Y_SLOT (src/verifier/Verifier.sol#60) is too similar to Verifier.VK_LOOKUP_TABLE_2_X_SLOT (src/verifier/Verifier.sol#61)
Variable Verifier.VK_LOOKUP_TABLE_2_X_SLOT (src/verifier/Verifier.sol#61) is too similar to Verifier.VK_LOOKUP_TABLE_3_X_SLOT (src/verifier/Verifier.sol#63)
Variable Verifier.VK_LOOKUP_TABLE_2_X_SLOT (src/verifier/Verifier.sol#61) is too similar to Verifier.VK_LOOKUP_TABLE_3_Y_SLOT (src/verifier/Verifier.sol#64)
Variable Verifier.VK_LOOKUP_TABLE_0_X_SLOT (src/verifier/Verifier.sol#57) is too similar to Verifier.VK_LOOKUP_TABLE_2_Y_SLOT (src/verifier/Verifier.sol#62)
Variable Verifier.VK_LOOKUP_TABLE_0_Y_SLOT (src/verifier/Verifier.sol#58) is too similar to Verifier.VK_LOOKUP_TABLE_2_Y_SLOT (src/verifier/Verifier.sol#62)
Variable Verifier.VK_LOOKUP_TABLE_1_X_SLOT (src/verifier/Verifier.sol#59) is too similar to Verifier.VK_LOOKUP_TABLE_2_Y_SLOT (src/verifier/Verifier.sol#62)
Variable Verifier.VK_LOOKUP_TABLE_1_Y_SLOT (src/verifier/Verifier.sol#60) is too similar to Verifier.VK_LOOKUP_TABLE_2_Y_SLOT (src/verifier/Verifier.sol#62)
Variable Verifier.VK_LOOKUP_TABLE_2_X_SLOT (src/verifier/Verifier.sol#61) is too similar to Verifier.VK_LOOKUP_TABLE_2_Y_SLOT (src/verifier/Verifier.sol#62)
Variable Verifier.VK_LOOKUP_TABLE_2_Y_SLOT (src/verifier/Verifier.sol#62) is too similar to Verifier.VK_LOOKUP_TABLE_3_X_SLOT (src/verifier/Verifier.sol#63)
Variable Verifier.VK_LOOKUP_TABLE_2_Y_SLOT (src/verifier/Verifier.sol#62) is too similar to Verifier.VK_LOOKUP_TABLE_3_Y_SLOT (src/verifier/Verifier.sol#64)
Variable Verifier.VK_LOOKUP_TABLE_0_X_SLOT (src/verifier/Verifier.sol#57) is too similar to Verifier.VK_LOOKUP_TABLE_3_X_SLOT (src/verifier/Verifier.sol#63)
Variable Verifier.VK_LOOKUP_TABLE_0_Y_SLOT (src/verifier/Verifier.sol#58) is too similar to Verifier.VK_LOOKUP_TABLE_3_X_SLOT (src/verifier/Verifier.sol#63)
Variable Verifier.VK_LOOKUP_TABLE_1_X_SLOT (src/verifier/Verifier.sol#59) is too similar to Verifier.VK_LOOKUP_TABLE_3_X_SLOT (src/verifier/Verifier.sol#63)
Variable Verifier.VK_LOOKUP_TABLE_1_Y_SLOT (src/verifier/Verifier.sol#60) is too similar to Verifier.VK_LOOKUP_TABLE_3_X_SLOT (src/verifier/Verifier.sol#63)
Variable Verifier.VK_LOOKUP_TABLE_3_X_SLOT (src/verifier/Verifier.sol#63) is too similar to Verifier.VK_LOOKUP_TABLE_3_Y_SLOT (src/verifier/Verifier.sol#64)
Variable Verifier.VK_LOOKUP_TABLE_0_X_SLOT (src/verifier/Verifier.sol#57) is too similar to Verifier.VK_LOOKUP_TABLE_3_Y_SLOT (src/verifier/Verifier.sol#64)
Variable Verifier.VK_LOOKUP_TABLE_0_Y_SLOT (src/verifier/Verifier.sol#58) is too similar to Verifier.VK_LOOKUP_TABLE_3_Y_SLOT (src/verifier/Verifier.sol#64)
Variable Verifier.VK_LOOKUP_TABLE_1_X_SLOT (src/verifier/Verifier.sol#59) is too similar to Verifier.VK_LOOKUP_TABLE_3_Y_SLOT (src/verifier/Verifier.sol#64)
Variable Verifier.VK_LOOKUP_TABLE_TYPE_X_SLOT (src/verifier/Verifier.sol#66) is too similar to Verifier.VK_LOOKUP_TABLE_TYPE_Y_SLOT (src/verifier/Verifier.sol#67)
Variable Verifier.VK_PERMUTATION_0_X_SLOT (src/verifier/Verifier.sol#45) is too similar to Verifier.VK_PERMUTATION_1_X_SLOT (src/verifier/Verifier.sol#47)
Variable Verifier.VK_PERMUTATION_0_X_SLOT (src/verifier/Verifier.sol#45) is too similar to Verifier.VK_PERMUTATION_3_X_SLOT (src/verifier/Verifier.sol#51)
Variable Verifier.VK_PERMUTATION_0_X_SLOT (src/verifier/Verifier.sol#45) is too similar to Verifier.VK_PERMUTATION_0_Y_SLOT (src/verifier/Verifier.sol#46)
Variable Verifier.VK_PERMUTATION_0_Y_SLOT (src/verifier/Verifier.sol#46) is too similar to Verifier.VK_PERMUTATION_1_X_SLOT (src/verifier/Verifier.sol#47)
Variable Verifier.VK_PERMUTATION_0_Y_SLOT (src/verifier/Verifier.sol#46) is too similar to Verifier.VK_PERMUTATION_3_X_SLOT (src/verifier/Verifier.sol#51)
Variable Verifier.VK_PERMUTATION_1_X_SLOT (src/verifier/Verifier.sol#47) is too similar to Verifier.VK_PERMUTATION_3_X_SLOT (src/verifier/Verifier.sol#51)
Variable Verifier.VK_PERMUTATION_0_X_SLOT (src/verifier/Verifier.sol#45) is too similar to Verifier.VK_PERMUTATION_1_Y_SLOT (src/verifier/Verifier.sol#48)
Variable Verifier.VK_PERMUTATION_0_Y_SLOT (src/verifier/Verifier.sol#46) is too similar to Verifier.VK_PERMUTATION_1_Y_SLOT (src/verifier/Verifier.sol#48)
Variable Verifier.VK_PERMUTATION_1_X_SLOT (src/verifier/Verifier.sol#47) is too similar to Verifier.VK_PERMUTATION_1_Y_SLOT (src/verifier/Verifier.sol#48)
Variable Verifier.VK_PERMUTATION_1_Y_SLOT (src/verifier/Verifier.sol#48) is too similar to Verifier.VK_PERMUTATION_2_X_SLOT (src/verifier/Verifier.sol#49)
Variable Verifier.VK_PERMUTATION_1_Y_SLOT (src/verifier/Verifier.sol#48) is too similar to Verifier.VK_PERMUTATION_2_Y_SLOT (src/verifier/Verifier.sol#50)
Variable Verifier.VK_PERMUTATION_1_Y_SLOT (src/verifier/Verifier.sol#48) is too similar to Verifier.VK_PERMUTATION_3_X_SLOT (src/verifier/Verifier.sol#51)
Variable Verifier.VK_PERMUTATION_1_Y_SLOT (src/verifier/Verifier.sol#48) is too similar to Verifier.VK_PERMUTATION_3_Y_SLOT (src/verifier/Verifier.sol#52)
Variable Verifier.VK_PERMUTATION_0_X_SLOT (src/verifier/Verifier.sol#45) is too similar to Verifier.VK_PERMUTATION_2_X_SLOT (src/verifier/Verifier.sol#49)
Variable Verifier.VK_PERMUTATION_0_Y_SLOT (src/verifier/Verifier.sol#46) is too similar to Verifier.VK_PERMUTATION_2_X_SLOT (src/verifier/Verifier.sol#49)
Variable Verifier.VK_PERMUTATION_1_X_SLOT (src/verifier/Verifier.sol#47) is too similar to Verifier.VK_PERMUTATION_2_X_SLOT (src/verifier/Verifier.sol#49)
Variable Verifier.VK_PERMUTATION_2_X_SLOT (src/verifier/Verifier.sol#49) is too similar to Verifier.VK_PERMUTATION_2_Y_SLOT (src/verifier/Verifier.sol#50)
Variable Verifier.VK_PERMUTATION_2_X_SLOT (src/verifier/Verifier.sol#49) is too similar to Verifier.VK_PERMUTATION_3_X_SLOT (src/verifier/Verifier.sol#51)
Variable Verifier.VK_PERMUTATION_2_X_SLOT (src/verifier/Verifier.sol#49) is too similar to Verifier.VK_PERMUTATION_3_Y_SLOT (src/verifier/Verifier.sol#52)
Variable Verifier.VK_PERMUTATION_0_X_SLOT (src/verifier/Verifier.sol#45) is too similar to Verifier.VK_PERMUTATION_2_Y_SLOT (src/verifier/Verifier.sol#50)
Variable Verifier.VK_PERMUTATION_0_Y_SLOT (src/verifier/Verifier.sol#46) is too similar to Verifier.VK_PERMUTATION_2_Y_SLOT (src/verifier/Verifier.sol#50)
Variable Verifier.VK_PERMUTATION_1_X_SLOT (src/verifier/Verifier.sol#47) is too similar to Verifier.VK_PERMUTATION_2_Y_SLOT (src/verifier/Verifier.sol#50)
Variable Verifier.VK_PERMUTATION_2_Y_SLOT (src/verifier/Verifier.sol#50) is too similar to Verifier.VK_PERMUTATION_3_X_SLOT (src/verifier/Verifier.sol#51)
Variable Verifier.VK_PERMUTATION_0_X_SLOT (src/verifier/Verifier.sol#45) is too similar to Verifier.VK_PERMUTATION_3_Y_SLOT (src/verifier/Verifier.sol#52)
Variable Verifier.VK_PERMUTATION_0_Y_SLOT (src/verifier/Verifier.sol#46) is too similar to Verifier.VK_PERMUTATION_3_Y_SLOT (src/verifier/Verifier.sol#52)
Variable Verifier.VK_PERMUTATION_1_X_SLOT (src/verifier/Verifier.sol#47) is too similar to Verifier.VK_PERMUTATION_3_Y_SLOT (src/verifier/Verifier.sol#52)
Variable Verifier.VK_PERMUTATION_2_Y_SLOT (src/verifier/Verifier.sol#50) is too similar to Verifier.VK_PERMUTATION_3_Y_SLOT (src/verifier/Verifier.sol#52)
Variable Verifier.VK_PERMUTATION_3_X_SLOT (src/verifier/Verifier.sol#51) is too similar to Verifier.VK_PERMUTATION_3_Y_SLOT (src/verifier/Verifier.sol#52)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Verifier.verify.asm_0.revertWithMessage() (src/verifier/Verifier.sol#347-358) uses literals with too many digits:
        - mstore(uint256,uint256)(0x00,0x08c379a000000000000000000000000000000000000000000000000000000000) (src/verifier/Verifier.sol#349)
Verifier.verify.asm_0.revertWithMessage() (src/verifier/Verifier.sol#347-358) uses literals with too many digits:
        - mstore(uint256,uint256)(0x04,0x0000000000000000000000000000000000000000000000000000000000000020) (src/verifier/Verifier.sol#351)
Verifier.slitherConstructorConstantVariables() (src/verifier/Verifier.sol#16-1672) uses literals with too many digits:
        - DOMAIN_SIZE = 0x4000000 (src/verifier/Verifier.sol#230)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
Verifier.QUERIES_AT_Z_1_Y_SLOT (src/verifier/Verifier.sol#181) is never used in Verifier (src/verifier/Verifier.sol#16-1672)
Verifier.AGGREGATED_AT_Z_X_SLOT (src/verifier/Verifier.sol#190) is never used in Verifier (src/verifier/Verifier.sol#16-1672)
Verifier.AGGREGATED_AT_Z_Y_SLOT (src/verifier/Verifier.sol#191) is never used in Verifier (src/verifier/Verifier.sol#16-1672)
Verifier.AGGREGATED_AT_Z_OMEGA_X_SLOT (src/verifier/Verifier.sol#193) is never used in Verifier (src/verifier/Verifier.sol#16-1672)
Verifier.AGGREGATED_AT_Z_OMEGA_Y_SLOT (src/verifier/Verifier.sol#194) is never used in Verifier (src/verifier/Verifier.sol#16-1672)
Verifier.AGGREGATED_OPENING_AT_Z_SLOT (src/verifier/Verifier.sol#196) is never used in Verifier (src/verifier/Verifier.sol#16-1672)
Verifier.AGGREGATED_OPENING_AT_Z_OMEGA_SLOT (src/verifier/Verifier.sol#197) is never used in Verifier (src/verifier/Verifier.sol#16-1672)
Verifier.PAIRING_BUFFER_POINT_Y_SLOT (src/verifier/Verifier.sol#204) is never used in Verifier (src/verifier/Verifier.sol#16-1672)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
```

- As a result of the tests carried out with the Slither tool, some results were obtained and reviewed by Halborn. Based on the results reviewed, the vulnerabilities were determined to be false positives.

5.2 AUTOMATED SECURITY SCAN

Description:

Halborn used automated security scanners to assist with detection of well-known security issues, and to identify low-hanging fruits on the targets for this engagement. Among the tools used was MythX, a security analysis service for Ethereum smart contracts. MythX performed a scan on the verifier contract and sent the compiled results to the analyzers to locate any vulnerabilities.

MythX results:

Report for Verifier.sol
https://dashboard.mythx.io/#/console/analyses/38dc9fe1-8e22-4008-a6a8-2c4a4e81bfcf

Line	SWC Title	Severity	Short Description
3	(SWC-103) Floating Pragma	Low	A floating pragma is set.

- No major issues found by Mythx.

THANK YOU FOR CHOOSING

// HALBORN