Passport ZK Circuits

Rarimo

HALBORN

Passport ZK Circuits - Rarimo

Prepared by: HALBORN

Last Updated 03/16/2024

Date of Engagement by: February 23rd, 2024 - March 8th, 2024

Summary

100% OF ALL REPORTED FINDINGS HAVE BEEN ADDRESSED

ALL FINDINGS	CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
2	0	0	0	1	1

TABLE OF CONTENTS

1. Introduction

The Rarimo team engaged Halborn to conduct a security assessment on their zero knowledge circuits and the verifier, beginning on 23/02/2024 and ending on 08/03/2024. The security assessment was scoped to the circuits provided to the Halborn team.

2. Assessment Summary

The team at Halborn was provided two weeks for the engagement and assigned a full-time security engineer to verify the security of the zero knowledge circuits and the verifier. The security engineer is a blockchain, smart-contract, and ZK security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this assessment is to achieve the following :

- Ensure that the circuits operate as intended.

- Identify potential security issues within the circuits.

- Ensure that the verifier operate as intended.

- Identify potential security issues within the verifier contracts.

In summary, Halborn identified some security risks that were successfully addressed by the Rarimo team.

3. Test Approach And 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 code and can quickly identify items that do not follow the security best practices. The following phases and associated tools were used during the assessment:

- Research into architecture and purpose.
- Manual code review and walkthrough.
- Graphing out functionality and contract logic/connectivity/functions.
- Manual testing by custom scripts.
- Used circomspect, a static analyzer and linter.
- Used Picus, which checks for under-constrained bugs.

Out-Of-Scope

- External libraries and financial-related attacks.
- New features/implementations after/with the **remediation commit IDs**.

4. 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.

4.1 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:

EXPLOITABILIY 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:

$$E = \prod m_e$$

4.2 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) Low (I:L) Medium (I:M) High (I:H) Critical (I:C)	0 0.25 0.5 0.75 1
Integrity (I)	None (I:N) Low (I:L) Medium (I:M) High (I:H) Critical (I:C)	0 0.25 0.5 0.75 1
Availability (A)	None (A:N) Low (A:L) Medium (A:M) High (A:H) Critical (A:C)	0 0.25 0.5 0.75 1

IMPACT METRIC (m_I)	METRIC VALUE	NUMERICAL VALUE
Deposit (D)	None (D:N) Low (D:L) Medium (D:M) High (D:H) Critical (D:C)	0 0.25 0.5 0.75 1
Yield (Y)	None (Y:N) Low (Y:L) Medium (Y:M) High (Y:H) Critical (Y:C)	0 0.25 0.5 0.75 1

Impact I is calculated using the following formula:

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

4.3 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.

METRICS:

SEVERITY 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

5. SCOPE

FILES AND REPOSITORY

(a) Repository: passport-zk-circuits

(b) Assessed Commit ID: 48793f9

(c) Contracts in scope:

- - dualMux.circom
- - hashLeftRight.circom
- - merkleTree.circom
- - dateComparison.circom
- - passportVerificationCore.circom
- - passportVerificationSHA1.circom
- - passportVerificationSHA256.circom
- - constants.circom
- - f.circom
- - parity.circom
- - rotate.circom
- - sha1.circom
- - sha1compression.circom
- - t.circom
- - xor4.circom
- photoVerifier.circom
- vote.circom

Out-of-Scope:

REMEDIATION COMMIT ID:

- f0e8d50
- eb5bb5d

Out-of-Scope: New features/implementations after the remediation commit IDs.

6. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	1	1

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL-05 - MISSING CONSTRAINT IN THE OUTPUT OF PASSPORTVERIFICATIONSHA1 CIRCUIT	Low	SOLVED - 03/14/2024
HAL-03 - DIFFERENT CIRCOM VERSIONS USED ALONG THE PROJECT	Informational	SOLVED - 03/13/2024

7. FINDINGS & TECH DETAILS

7.1 (HAL-05) MISSING CONSTRAINT IN THE OUTPUT OF `PASSPORTVERIFICATIONSHA1` CIRCUIT

// LOW

Description

The output signal of the `passportVerificationSHA1` circuit is formed by the following elements:

- `out[0]` -> SHA1 hash of the `in` input signal.
- `out[1]` -> not used.
- `out[2]` -> number representation of the passport issuer code.

Both `out[0]` and `out[2]` are properly constrained, but `out[1]` is not constrained in any way. That makes it possible for the prover to put whatever value he wants in there. This may not pose a serious risk, but It is recommended constraining `out[1]` to be zero:

```
passportVerificationSHA1, line 50template PassportVerificationSHA1(N) {

    ...

    out[0] <== bits2NumHash.out;
+   out[1] <== 0;
}
```

BVSS

AO:A/AC:L/AX:L/C:L/I:N/A:N/D:N/Y:N/R:N/S:C (3.1)

Recommendation

It is recommended to constrain **out[1]** to be zero: [**passportVerificationSHA1, line 50**] (https://github.com/rarimo/passport-zk-circuits/blob/48793f924383cd11d4c3426b4697c2f9b4efa60f/passportVerification/passportVerificationSHA1.circom#L50)

```
template PassportVerificationSHA1(N) {

    ...

    out[0] <== bits2NumHash.out;
+   out[1] <== 0;
}
```

Remediation Plan

SOLVED : The Rarimo team solved the issue by implementing the provided suggestion.

Remediation Hash

https://github.com/rarimo/passport-zk-circuits/commit/f0e8d5096c26d425571a25c1ff83f1eff5773dac#diff-26241884654ea3527f948b3332d3f586fb63fd7815a1ea6c4c7ec1267ba59242

7.2 (HAL-03) DIFFERENT CIRCOM VERSIONS USED ALONG THE PROJECT

// INFORMATIONAL

Description

During the code review of the one circuit, namely the `SMTVerifier` circuit, uses a different version of the Circom compiler (`2.0.0.`) instead of the one the whole project uses (`2.1.6`). Using the same version for all circuits is recommended, as older compiler versions may have bugs that recent releases may have fixed.

Score

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:N/R:N/S:C (0.0)

Recommendation

Using the same version for all circuits is recommended, as older compiler versions may have bugs that recent releases may have fixed.

Remediation Plan

SOLVED : The Rarimo team solved the issue by implementing the provided suggestion.

Remediation Hash

https://github.com/rarimo/passport-zk-circuits/commit/eb5bb5dd5522533c443e7c4726ea725eda81ec3e#diff-5bd3302d8df02c76860fef4213bf2d8422b29e0a23e685212a410978aa06ad5e

8. AUTOMATED TESTING

Halborn used automated testing techniques to enhance coverage of certain areas of the scoped component. Among the tools used were **circumspect and picus**. After Halborn verified all the code and scoped structures in the repository and could compile them correctly, these tools were leveraged on scoped structures. With these tools, Halborn can statically verify security-related issues across the entire codebase.

```
root@02bd20ad49d3:/Picus# ./run-picus ./passport-zk-circuits/voteSMT/voteSMT.circom
The circuit is properly constrained
Exiting Picus with the code 8
```

```
root@02bd20ad49d3:/Picus# ./run-picus ./passport-zk-circuits/vote/vote.circom
The circuit is properly constrained
Exiting Picus with the code 8
```

```
root@02bd20ad49d3:/Picus# ./run-picus ./passport-zk-circuits/voteSMT/voteSMT2.circom
The circuit is properly constrained
Exiting Picus with the code 8
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/utils/constants.circom
 circomspect: analyzing template 'K'
 circomspect: analyzing template 'H'
 circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/dateComparison.circom
circomspect: analyzing template 'DateIsLess'
warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/dateComparison.circom:24:25
    │
24  │         monthLess.in[1] <== secondMonth;
    │                             ^^^^^^^^^^^ `secondMonth` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/dateComparison.circom:18:24
    │
18  │         yearLess.in[0] <== firstYear;
    │                            ^^^^^^^^^ `firstYear` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/dateComparison.circom:19:24
    │
19  │         yearLess.in[1] <== secondYear;
    │                            ^^^^^^^^^^ `secondYear` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/dateComparison.circom:23:25
    │
23  │         monthLess.in[0] <== firstMonth;
    │                             ^^^^^^^^^^ `firstMonth` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/dateComparison.circom:28:23
    │
28  │         dayLess.in[0] <== firstDay;
    │                           ^^^^^^^^ `firstDay` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/dateComparison.circom:29:23
    │
29  │         dayLess.in[1] <== secondDay;
    │                           ^^^^^^^^^ `secondDay` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

circomspect: 6 issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./merkleTree/dualMux.circom
 circomspect: analyzing template 'DualMux'
 circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/utils/f.circom
 circomspect: analyzing template 'f_t'
 circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./merkleTree/hashLeftRight.circom
circomspect: analyzing template 'HashLeftRight'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./merkleTree/merkleTree.circom
circomspect: analyzing template 'MerkleTreeVerifier'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/utils/parity.circom
circomspect: analyzing template 'Parity_t'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/passportVerificationCore.circom
circomspect: analyzing template 'ValidateMonthCorrectness'
warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/passportVerificationCore.circom:13:37
    │
 13 │         monthLessThanThirteen.in[0] <== month;
    │                                         ^^^^^ `month` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/passportVerificationCore.circom:14:37
    │
 14 │         monthLessThanThirteen.in[1] <== 13;
    │                                         ^^ `13` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

circomspect: analyzing template 'PassportVerificationCore'
warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/passportVerificationCore.circom:120:29
    │
120 │         isPrevCentury.in[0] <== currDateYear;
    │                                 ^^^^^^^^^^^ `currDateYear` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/passportVerificationCore.circom:121:29
    │
121 │         isPrevCentury.in[1] <== birthYear;
    │                                 ^^^^^^^^^ `birthYear` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

circomspect: 4 issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/passportVerificationSHA1.circom
circomspect: analyzing template 'PassportVerificationSHA1'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/passportVerificationSHA256.circom
circomspect: analyzing template 'PassportVerificationSHA256'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/utils/rotate.circom
circomspect: analyzing template 'RotL'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/utils/sha1.circom
circomspect: analyzing template 'Sha1'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/utils/sha1compression.c
circomspect: analyzing template 'Sha1compression'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./voteSMT/SMTVerifier.circom
circomspect: analyzing template 'SMTLevIns'
circomspect: analyzing template 'SMTVerifierLevel'
circomspect: analyzing template 'SMTVerifier'
circomspect: analyzing template 'SMTVerifierSM'
circomspect: analyzing template 'SMTHash1'
circomspect: analyzing template 'SMTHash2'
circomspect: analyzing template 'CommitmentHasher'
circomspect: No issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/utils/t.circom
circomspect: analyzing template 'T'
warning: Using the signal assignment operator `<--` does not constrain the assigned signal.
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/utils/t.circom:52:6
    │
 52 │       sum_modulo <-- sum.out % 2**32;
    │       ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ The assigned signal `sum_modulo` is not constrained here.
    ·
 55 │       less_than.in[0] <== sum_modulo;
    │       ------------------------------ The signal `sum_modulo` is constrained here.
    ·
 58 │       sum.out === quotient * 2**32 + sum_modulo;
    │       ------------------------------------------ The signal `sum_modulo` is constrained here.
    ·
 63 │       sum_binary_modulo.in <== sum_modulo;
    │       ---------------------------------- The signal `sum_modulo` is constrained here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#signal-assignment.

warning: Using the signal assignment operator `<--` does not constrain the assigned signal.
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/utils/t.circom:53:6
    │
 53 │       quotient <-- sum.out \ 2**32;
    │       ^^^^^^^^^^^^^^^^^^^^^^^^^^^ The assigned signal `quotient` is not constrained here.
    ·
 58 │       sum.out === quotient * 2**32 + sum_modulo;
    │       ------------------------------------------ The signal `quotient` is constrained here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#signal-assignment.

warning: Inputs to `LessThan` need to be constrained to ensure that they are non-negative
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/utils/t.circom:56:26
    │
 56 │       less_than.in[1] <== 2**32;
    │                           ^^^^^ `(2 ** 32)` needs to be constrained to ensure that it is <= p/2.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unconstrained-less-than.

warning: Intermediate signals should typically occur in at least two separate constraints.
    ┌─ /home/neth/Desktop/passport-zk-circuits/passportVerification/utils/t.circom:49:6
    │
 49 │       signal quotient;
    │       ^^^^^^^^^^^^^^^ The intermediate signal `quotient` is declared here.
    ·
 58 │       sum.out === quotient * 2**32 + sum_modulo;
    │       ------------------------------------------ The intermediate signal `quotient` is constrained here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#under-constrained-signal.

circomspect: 4 issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./vote/vote.circom
circomspect: analyzing template 'Vote'
warning: Intermediate signals should typically occur in at least two separate constraints.
    ┌─ /home/neth/Desktop/passport-zk-circuits/vote/vote.circom:55:5
    │
 55 │       signal voteSquare;
    │       ^^^^^^^^^^^^^^^^^ The intermediate signal `voteSquare` is declared here.
 56 │       voteSquare <== vote * vote;
    │       -------------------------- The intermediate signal `voteSquare` is constrained here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#under-constrained-signal.

circomspect: analyzing template 'CommitmentHasher'
circomspect: 1 issue found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./voteSMT/voteSMT.circom
circomspect: analyzing template 'VoteSMT'
warning: Intermediate signals should typically occur in at least two separate constraints.
    ┌─ /home/neth/Desktop/passport-zk-circuits/voteSMT/voteSMT.circom:37:5
    │
 37 │       signal voteSquare <== vote * vote;
    │       ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
    │       |
    │       The intermediate signal `voteSquare` is declared here.
    │       The intermediate signal `voteSquare` is constrained here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#under-constrained-signal.

warning: Intermediate signals should typically occur in at least two separate constraints.
    ┌─ /home/neth/Desktop/passport-zk-circuits/voteSMT/voteSMT.circom:38:5
    │
 38 │       signal votingAddressSquare <== votingAddress * votingAddress;
    │       ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
    │       |
    │       The intermediate signal `votingAddressSquare` is declared here.
    │       The intermediate signal `votingAddressSquare` is constrained here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/ma

circomspect: 2 issues found.
```

The terminal process "/usr/bin/bash" terminated with exit code: 1.

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./voteSMT/voteSMT2.circom
circomspect: analyzing template 'VoteSMT2'
warning: Intermediate signals should typically occur in at least two separate constraints.
    ┌─ /home/neth/Desktop/passport-zk-circuits/voteSMT/voteSMT2.circom:57:5
    │
 57 │     signal votingAddressSquare <== votingAddress * votingAddress;
    │     ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
    │     |
    │     The intermediate signal `votingAddressSquare` is declared here.
    │     The intermediate signal `votingAddressSquare` is constrained here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#under-constrained-signal.

warning: Intermediate signals should typically occur in at least two separate constraints.
    ┌─ /home/neth/Desktop/passport-zk-circuits/voteSMT/voteSMT2.circom:56:5
    │
 56 │     signal voteSquare <== vote * vote;
    │     ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
    │     |
    │     The intermediate signal `voteSquare` is declared here.
    │     The intermediate signal `voteSquare` is constrained here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#under-constrained-signal.

warning: The output signal `nullifierHash` defined by the template `SMTVerifier` is not constrained in `VoteSMT2`.
    ┌─ /home/neth/Desktop/passport-zk-circuits/voteSMT/voteSMT2.circom:31:30
    │
 31 │     component smtVerifier2 = SMTVerifier(treeDepth);
    │                              ^^^^^^^^^^^^^^^^^^^^^^ The template `SMTVerifier` is instantiated here.
    │
    = For more details, see https://github.com/trailofbits/circomspect/blob/main/doc/analysis_passes.md#unused-output-signal.

circomspect: 3 issues found.
```

```
neth@neth:~/Desktop/passport-zk-circuits$ circomspect ./passportVerification/utils/xor4.circom
circomspect: analyzing template 'Xor4'
circomspect: No issues found.
```

Halborn strongly recommends conducting a follow-up assessment of the project either within six months or immediately following any material changes to the codebase, whichever comes first. This approach is crucial for maintaining the project's integrity and addressing potential vulnerabilities introduced by code modifications.