SALUS SECURITY

AUG 2025

CODE SECURITY ASSESSMENT

XONE

Overview

Project Summary

- Name: Xone- DID contracts
- Platform: EVM-compatible chains
- Language: Solidity
- Repository:
    - https://github.com/xone-name/did_contracts
- Audit Range: See Appendix - 1

Project Dashboard

Application Summary

Name	Xone - DID contracts
Version	v3
Type	Solidity
Dates	Aug 20 2025
Logs	Aug 15 2025; Aug 19 2025; Aug 20 2025

Vulnerability Summary

Total High-Severity issues	0
Total Medium-Severity issues	0
Total Low-Severity issues	4
Total informational issues	1
Total	5

Contact

E-mail: support@salusec.io

Risk Level Description

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for clients' reputations or serious financial implications for clients and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental to the client's reputation if exploited, or is reasonably likely to lead to a moderate financial impact.
Low Risk	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
Informational	The issue does not pose an immediate risk, but is relevant to security best practices or defense in depth.

Content

Introduction

1.1 About SALUS

At Salus Security, we are in the business of trust.

We are dedicated to tackling the toughest security challenges facing the industry today. By building foundational trust in technology and infrastructure through security, we help clients to lead their respective industries and unlock their full Web3 potential.

Our team of security experts employ industry-leading proof-of-concept (PoC) methodology for demonstrating smart contract vulnerabilities, coupled with advanced red teaming capabilities and a stereoscopic vulnerability detection service, to deliver comprehensive security assessments that allow clients to stay ahead of the curve.

In addition to smart contract audits and red teaming, our Rapid Detection Service for smart contracts aims to make security accessible to all. This high calibre, yet cost-efficient, security tool has been designed to support a wide range of business needs including investment due diligence, security and code quality assessments, and code optimisation.

We are reachable on Telegram (https://t.me/salusec), Twitter (https://twitter.com/salus_sec), or Email (support@salusec.io).

1.2 Audit Breakdown

The objective was to evaluate the repository for security-related issues, code quality, and adherence to specifications and best practices. Possible issues we looked for included (but are not limited to):

- Risky external calls
- Integer overflow/underflow
- Transaction-ordering dependence
- Timestamp dependence
- Access control
- Call stack limits and mishandled exceptions
- Number rounding errors
- Centralization of power
- Logical oversights and denial of service
- Business logic specification
- Code clones, functionality duplication

1.3 Disclaimer

Note that this security audit is not designed to replace functional tests required before any software release and does not give any warranties on finding all possible security issues with the given smart contract(s) or blockchain software, i.e., the evaluation result does not guarantee the nonexistence of any further findings of security issues.

Release Notes

Based on our review, the development of xone-name did-contract is entirely based on the ENS protocol, with no modifications made to the core logic. As a result, the security of the code also relies on that of the ENS protocol. We encourage the team to regularly monitor the status of third-party dependencies to mitigate potential impacts in case they fail or become unreliable.

Module Path	Description (optional)	Original Source Link
ens-contracts/contracts	Core logic of the ens	ENS V1.4.0

Findings

2.1 Summary of Findings

ID	Title	Severity	Category	Status
1	Missing events for setKeeper() function	Low	Logging	Resolved
2	Missing minimum duration and name Length validation in register()	Low	Data Validation	Resolved
3	Third-party dependencies	Low	Dependency	Acknowledged
4	Missing Test Suite for XoneRegistrarController	Low	Testing	Acknowledged
5	Redundant Code	Informational	Redundancy	Resolved

2.2 Notable Findings

Significant flaws that impact system confidentiality, integrity, or availability are listed below.

1. Missing events for setKeeper() function

Severity: Low	Category: Logging
Target: - contracts/didregistrar/XoneRegistrarController.sol	

Description

Events allow capturing the changed parameters so that off-chain tools/interfaces can register such changes that allow users to evaluate them. Missing events do not promote transparency and if such changes immediately affect users' perception of fairness or trustworthiness, they could exit the protocol causing a reduction in protocol users.

In the `XoneRegistrarController` contract, the event is lacking in the `setKeeper()` function.

contracts/didregistrar/XoneRegistrarController.sol:L103-L105

```
function setKeeper(address keeper, bool status) external onlyOwner {
    keepers[keeper] = status;
}
```

Recommendation

It is recommended to emit events for the `setKeeper()` function.

Status

The team has resolved this issue in commit 54bbddf.

2. Missing minimum duration and name Length validation in register()

Severity: Low	Category: Data Validation
Target: - contracts/didregistrar/XoneRegistrarController.sol	

Description

The `register()` function does not enforce the minimum registration duration (`MIN_REGISTRATION_DURATION`) or the minimum name length check (`valid(name)`) defined in the contract. The correct flow should enforce these checks inside `_consumeCommitment()`, but in this implementation, the commitment logic is commented out, and these validations are never executed.

contracts/didregistrar/XoneRegistrarController.sol:L160-L220

```
function register(
    string calldata name,
    address owner,
    uint256 duration,
    bytes32 secret,
    address resolver,
    bytes[] calldata data,
    bool reverseRecord,
    uint16 ownerControlledFuses
) public payable override onlyKeeper {
    ...
    // _consumeCommitment(
    //     name,
    //     duration,
    //     makeCommitment(
    //         name,
    //         owner,
    //         duration,
    //         secret,
    //         resolver,
    //         data,
    //         reverseRecord,
    //         ownerControlledFuses
    //     )
    // );
```

Recommendation

Add explicit validation in the `register()` function.

Status

The team has resolved this issue in commit 54bbddf.

3. Third-party dependencies

Severity: Low	Category: Dependency
Target: - contracts/didregistrar/XoneRegistrarController.sol	

Description

The `XoneRegistrarController` contract relies on the ENS contract to enable the basic functionality of the controller. The current audit treats third-party entities as black boxes and assumes they are working correctly. However, in reality, third parties could be compromised, resulting in the disruption of token functionalities.

Recommendation

We understand that the business logic requires interaction with the third parties. We encourage the team to regularly monitor the statuses of third parties to reduce the impacts when they are not functioning properly.

Status

This issue has been acknowledged by the team

4. Missing Test Suite for XoneRegistrarController

Severity: Low	Category: Testing
Target: - contracts/didregistrar/XoneRegistrarController.sol	

Description

A test suite to confirm the proper operation of the deployed contracts and crucial features like registering and admin role assignment is absent from the repository. Contract security and dependability are at risk because defects or regressions might be undiscovered. Test scripts with filenames ending in.ts are often located in the test/ directory in Harhat-based projects.

Recommendation

Think about leveraging Harhat to create a thorough test suite that covers essential functions. Ensuring contract correctness and promoting safe continuous development require a strong test suite.

Status

This issue has been acknowledged by the team

2.3 Informational Findings

5. Redundant Code

Severity: Informational	Category: Redundancy
Target: - contracts/didregistrar/XoneRegistrarController.sol	

Description

Unused code should be removed before deploying the contract to mainnet. We have identified the following functions are not being utilized:

contracts/didregistrar/XoneRegistrarController.sol:L255-L279

```
function _consumeCommitment(
    string memory name,
    uint256 duration,
    bytes32 commitment
) internal {
    ...
}
```

Recommendation

Consider removing the redundant code.

Status

The team has resolved this issue in commit 54bbddf.

Appendix

Appendix 1 - Files in Scope

This audit covered the following files in commit 880a3f0:

File	SHA-1 hash
contracts/didregistrar/XoneRegistrarController.sol	f2e4c25785ba9dfab81d7a319dc122af9255003f