



SMART CONTRACT SECURITY AUDIT

AXIR

Airdrop Contract

CLIENT

AXIR

LANGUAGE

Solidity

DOCUMENT

Document information

Project	AXIR, Airdrop Contract
Client	AXIR
Lead auditor	Piyush Shukla
Platform	Not recorded in the original report
Language	Solidity
Repository	Not recorded in the original report
Commit	Not recorded in the original report
Methods	Not recorded in the original report
Report date	Not recorded in the original report

Document history

VERSION	DATE	CHANGE
1.0	Not recorded	Original report issued View original report
1.1	October 2026	Reformatted into the Safe Edges standard template. Findings unchanged

This report was reformatted into the Safe Edges standard template in October 2026. Findings, severities, recommendations and code references are unchanged from the original report, which is preserved unchanged in the [Safe Edges Audit Portfolio on GitHub](#).

CONTENTS

Contents

01	Executive summary	3
02	Scope	4
03	Methodology	5
04	Severity and status	6
05	Findings summary	7
06	Detailed findings	8
	H-01 incorrect Modifier in emergencyWithdraw Function	8
	M-01 LACK OF DISABLEINITIALIZERS	8
	G-01 use ++i instead i++ in function airdropToken loop	8
	G-02 There is no need to initialize i to its default value in function airdropToken loop	9
	Disclaimer	10

Executive summary

About AXIR

AXIR's airdrop contract distributes tokens to recipients in batches and includes pause controls and an emergency withdrawal function.

Summary of the review

Safe Edges reviewed AXIR's airdrop contract. Four issues were identified: one High, one Medium and two Low gas optimisations. The High issue was a wrong modifier on emergencyWithdraw that blocked emergency withdrawals while the contract was paused. AXIR fixed the High and both gas issues and acknowledged the Medium.

Findings by severity



Findings by status

STATUS	ISSUES
● Resolved	3
● Acknowledged	1

Scope

Repository	Not recorded in the original report
Commit	Not recorded in the original report
Platform	Not recorded in the original report
Language	Solidity

IN-SCOPE FILES

Airdrop contract

Only the code at the commit above was reviewed. Changes made after this commit are not covered by this report.

Methodology

Methods recorded for this engagement: Not recorded in the original report.

Manual review

Auditors read the in-scope code line by line, tracing control flow, state changes and trust boundaries against the intended behaviour described by the client.

Threat modelling

We map privileged roles, external dependencies and value flows, then test the assumptions the code makes about each of them.

Functional and automated testing

Where tooling exists for the language and chain, we run static analysis and targeted tests to confirm suspected issues and check edge cases.

Fix review

After the client addresses the findings, we review the updated code and record the final status of each issue.

Severity and status

Severity levels

SEVERITY	DEFINITION
Critical	Direct loss or permanent lock of funds, or full compromise of the protocol, with few or no preconditions. Must be fixed before deployment.
High	Significant loss of funds, broken core functionality or manipulation of critical state under realistic conditions. Should be fixed before deployment.
Medium	Limited loss or degraded functionality, or issues that need specific conditions or privileged access to exploit. Should be fixed.
Low	Minor impact, low likelihood, or deviations from best practice. Fix when practical.
Informational	Code quality, documentation, gas and style observations with no direct security impact.

Status definitions

STATUS	DEFINITION
Resolved	The issue was fixed and the fix was verified.
Partially resolved	The risk was reduced but not fully removed.
Acknowledged	The client accepted the issue without changing the code.
Open	The issue was not addressed at the time of the report.
Not recorded	The original report did not record a resolution status for this issue.

Findings summary

ID	TITLE	SEVERITY	STATUS
H-01	incorrect Modifier in emergencyWithdraw Function	High	● Resolved
M-01	LACK OF DISABLEINITIALIZERS	Medium	● Acknowledged
G-01	use ++i instead i++ in function airdropToken loop	Low	● Resolved
G-02	There is no need to initialize i to its default value in function airdropToken loop	Low	● Resolved

Detailed findings

H-01 incorrect Modifier in emergencyWithdraw Function

Severity **High** Status ● Resolved

DESCRIPTION

Airdrop contract, there is a discrepancy in the modifier used for the emergencyWithdraw function. The modifier used is whenNotPaused, which allows the function to be executed when the contract is not paused. However, the function is intended to be used for emergency withdrawals when contract is paused, or you should be allowed even when the contract is paused and not paused

MITIGATION

- Replace modifier Whennotpaused modifier with Whenpaused
- or
- Remove both modifier

M-01 LACK OF DISABLEINITIALIZERS

Severity **Medium** Status ● Acknowledged

DESCRIPTION

Attacker can re-initialize the function you must be add `_disableInitializers()`

MITIGATION

```
constructor() { _disableInitializers(); }
```

G-01 use ++i instead i++ in function airdropToken loop

Severity **Low** Status ● Resolved Category Gas

DESCRIPTION

use ++i instead i++

G-02 There is no need to initialize i to its default value in function airdropToken loop

Severity Low Status ● Resolved Category Gas

DESCRIPTION

`uint i;` instead `uint i=0;`

Disclaimer

This report is based on the code at the commit listed in the Scope section and on information the client provided during the review. It reflects Safe Edges' assessment at the time of the review and is not a guarantee that the code is free of vulnerabilities.

Changes made after the reviewed commit, deployment configuration, off-chain components and third-party dependencies are outside the scope of this report unless stated otherwise.

Security is ongoing. We recommend continued testing, monitoring and bug bounty coverage alongside any audit.

This report is not investment advice and is not an endorsement of the project, its team or any token.



Safe Edges is a Web3 security firm and official security partner for the Fuel, ENI and XI EcoChain ecosystems. We work across smart contract audits, Rust and WASM reviews, ZK circuit review, AI and MCP security, and penetration testing.

200+

PROJECTS AUDITED

\$4B+

TVL SECURED

0

POST-AUDIT HACKS