safepress

Smart Contract Audit

For smart contract vulnerabilities, security exploits and attack vectors safepress

Code Review And Security Report

Important: This document likely contains critical information about the Client's software and hardware systems, security susceptibilities, descriptions of possible exploits and attack vectors. The document shall remain undisclosed until any significant vulnerabilities are remedied.

CLIENT:	XP.NETWORK
START DATE:	22nd August 2022
END DATE:	29th August 2022
TYPE, SUBTYPE:	Bridge

Scope

Repository:	https://github.com/XP-NETWORK/algorand-asa-bridge/
Commit hashes:	e5cbe1762ffa31e634663e9ead970b95f27496d3 (incl.)
Documentation:	No documentation
Tests:	Passing
Auditors:	Vladislav, Andrew
Review & Approval:	Sam
Smart Contract Audited:	xpnet / contracts.py

Definitions of vulnerability classification

Severity	Definition
Critical	Bug / Logic failures in the code that cause loss of assets / data manipulation.
High	Difficult to exploit problems which could result in elevated privileges, data loss etc.
Medium	Bug / Logic failures in the code which need to be fixed but cannot lead to loss of assets / data manipulation.
Low	Mostly related to unused code, style guide violations, code snippets with low effect etc.

The smart contracts were found with the following vulnerabilities:

Critical Vulnerabilities

- **No verification for logicSig Program.**

  Function (s): on_receive_foreign, on_receive_native, on_update_gk, on_pause, on_unpause, on_withdraw_fees.

  Description: There is no check in the contract to ensure the validity of the compiled teal program that is used to create the logicSig.

  Recommendation: Add assert to check the validity of the logicSig compiled teal program which is supposed to be equal to Txn.sender()**.**

  Status: Done

High-level Vulnerabilities

- **Remove unnecessary idle transactions.**

  Function (s): on_receive_foreign, on_receive_native, on_update_gk, on_pause, on_unpause, on_withdraw_fees

  Description: NoOp inner transactions are being used throughout the contract which makes the code difficult to follow and maintain.

  Recommendation: Use OpUp budget increase utility for better code readability and maintenance**.**

  Status: Done

Medium-level Vulnerabilities

None

Low-level Vulnerabilities

- **Use of magic numbers.**

  Function (s): on_receive_foreign, on_receive_native, on_update_gk, on_pause, on_unpause, on_withdraw_fees,

  Description: Magic numbers are numbers without context and meaning. They make the code difficult to understand and read. For example “300_000” is used throughout the contract without any context of what this number means and why it is 300_000 and not any other number.

  Recommendation: Assign all magic numbers to descriptive variables that explains the context of the number.

  Status: Done

Informational

None

Executive Summary

Based on the audit findings the Client's contracts are: Not Secure

Not Secure	**Insufficiently Secured**	**Secured**	**Well Secured**

Disclaimers

SafePress Disclaimer
The smart contracts given for audit have been analyzed by the best industry practices at the date of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions). The audit makes no statements or warranties on the security of the code. It also cannot be considered a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

Technical Disclaimer
Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the audit cannot guarantee the explicit security of the audited smart contracts.