HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Xtra
Date: December 8th, 2021

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for Xtra.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	ERC20 token; Vesting; Staking; Investing
Platform	Binance Smart Chain / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/xtra-fund/xtra-contracts
Commit	62704775995b89ab7930efa14d4ba9148f99945d
Technical Documentation	YES
JS tests	NO
Website	xtra.fund
Timeline	29 NOVEMBER 2021 - 08 DECEMBER 2021
Changelog	07 DECEMBER 2021 - INITIAL AUDIT 08 DECEMBER 2021 - SECOND REVIEW

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by Xtra (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contract and its code review conducted between November 29th, 2021 - December 7th, 2021.

Second review conducted on December 8th, 2021.

Scope

The scope of the project is smart contracts in the repository:

Repository:
https://github.com/xtra-fund/xtra-contracts
Commit:
62704775995b89ab7930efa14d4ba9148f99945d
Technical Documentation: Yes
(https://drive.google.com/file/d/1gvo0pAtXWERefm4j2kcs5jezD-Yu4-VU/view)
JS tests: No
Contracts:
interfaces/IPancakeFactory.sol
interfaces/IPancakePair.sol
interfaces/IPancakeRouter01.sol
interfaces/IPancakeRouter02.sol
mocks/AllocationToken.sol
mocks/SimpleERC20.sol
others/Migrations.sol
xtra.sol
XtraInvesting.sol
XtraStaking.sol
XtraVesting.sol

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

Category	Check Item
Code review	▪ Reentrancy ▪ Ownership Takeover ▪ Timestamp Dependence ▪ Gas Limit and Loops ▪ DoS with (Unexpected) Throw ▪ DoS with Block Gas Limit ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ Costly Loop ▪ ERC20 API violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency ▪ Data Consistency
Functional review	▪ Business Logics Review ▪ Functionality Checks ▪ Access Control & Authorization ▪ Escrow manipulation ▪ Token Supply manipulation ▪ Assets integrity ▪ User Balances manipulation ▪ Data Consistency manipulation ▪ Kill-Switch Mechanism ▪ Operation Trails & Event Generation

Executive Summary

According to the assessment, the Customer's smart contracts are secured.

Insecure	Poor secured	Secured	Well-secured

You are here ↑

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Our team performed an analysis of code functionality, manual audit, and automated checks with Mythril and Slither. All issues found during automated analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. All found issues can be found in the Audit overview section.

As a result of the audit, security engineers found **4** medium and **3** low severity issues

After the second review security engineers found a slightly changed contact that now doesn't mint tokens each time, but has a pre-minted amount of tokens on the contract's address. Also, **1** medium and **1** low severity issues were found.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Audit overview

Critical

No critical issues were found.

High

No high severity issues were found.

Medium

1. No tests were provided

It’s recommended to cover all non-trivial contracts with tests.

The recommended coverage is a minimum of 95% for branches, while it should be definitely 100% for the main logic contracts.

2. Tautology or contradiction.

While uint256 is mean “unsigned integer 256 bits”, it is incorrect to check if it’s not less than zero, because it cannot be less than zero in any way.

Contracts: xtra.sol

Functions: distributeLPTokens, distributeLoanFund, activateInvestitions, activateAllocation

Recommendation: Fix the incorrect comparison by changing the value type or the comparison.

Status: Fixed

3. Allocation could be activated before staking started

The documentation is saying: “Can be launched only after the “Staking Start Date”. The code of the function “activateAllocation” doesn’t have a check for a staking already started.

Contracts: xtra.sol

Functions: activateAllocation

Recommendation: Add require “_stakingStartDate < block.timestamp”.

Status: Fixed

4. Staking/Unstake/Liquidation shouldn’t be called by contracts

The documentation is saying: “It cannot be triggered by other smart contracts.” in the “Staking”, “Unstake” and “Liquidation” sections.

The contract has a checking like: “msg.sender == tx.origin”, but this is not preventing contracts to call the function, because if the function would be called with delegate call, the check may pass.

Contracts: xtra.sol

Functions: stake, unstake

Recommendation: Add check for “msg.sender” to be not a contract.

Status: Fixed

Low

1. State variables that could be declared immutable.

Constant state variables that are initialized in the constructor should be declared immutable to save gas

Contracts: xtra.sol, XtraVesting.sol

Variables: _allocationTokenAddress, _pancakeFactoryAddress, _stableCoinAddress, _vestingLastDate

Recommendation: Add the **immutable** attribute to state variables that never change and are initialized in the constructor.

Status: Fixed

2. State variables that could be declared constant

Constant state variables should be declared constant to save gas.

Contracts: xtra.sol, XtraStaking.sol

Variables: _initialTokenPrice, DAYS_LIMIT_1, DAYS_LIMIT_2, DAYS_LIMIT_MAX, PERC_LIMIT_1, PERC_LIMIT_2

Recommendation: Add the **constant** attribute to state variables that never change.

Status: Fixed

3. Too many digits

Literals with many digits are difficult to read and review

Contracts: xtra.sol, XtraInvesting.sol, XtraStaking.sol

Functions: XtraInvesting._addInvestors, XtraStaking._longerDurationBonus, Xtra.slitherConstructorVariables, Xtra.slitherConstructorVariables, Xtra.constructor

Recommendation: Please consider using scientific notation with the ether unit suffix (for example: "*2e9 ether*" instead of "*2000000000 * 10**18*").

Status: Partly fixed. Still have in the Xtra.constructor

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools.

The audit report contains all found security vulnerabilities and other issues in the reviewed code.

As a result of the audit, security engineers found **4** medium and **3** low severity issues

After the second review security engineers found a slightly changed contact that now doesn't mint tokens each time, but has a pre-minted amount of tokens on the contract's address. Also, **1** medium and **1** low severity issues were found.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to 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 as a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other statements of the contract. 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 can't guarantee the explicit security of the audited smart contracts.