AlienX Security Review

Conducted by: **Georgi Trachev**

Report Version 1.0
October 16, 2024

Contents

1. About Georgi Trachev

Georgi Trachev is a smart contract security researcher. Throughout his experience he has uncovered numerous critical vulnerabilities in a wide variety of protocols. Reach out on X here.

2. Disclaimer

Smart contract security reviews are a time, resource and expertise bound effort where security researchers do their utmost to find as many vulnerabilities as they can. They are not a 100% guarantee of the security of smart contracts.

3. About AlienX

AlienX is a hyper-deflationary perpetual compounding system. It has mechanisms for burning and minting tokens, where the cost of minting AlienX tokens grows over an 8-week period.

4. Risk Classification

Severity	**Impact: High**	**Impact: Medium**	**Impact: Low**
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4.1 Impact

High: Causes a significant loss of funds, the protocol is directly compromised

Medium: A minor amount of funds at loss, or a functionality of the protocol can be affected

Low: Unexpected behaviour in the protocol, but does not present a significant harm

4.2 Likelihood

High: Direct attack vector; cost of attack negligible, compared to funds at risk

Medium: A sophisticated attack vector that is incentivized only in some conditions but is still likely to occur

Low: Requires many unlikely assumptions or the attacker is not incentivized

4.3 Actions required by security level

High: Issue **MUST** be fixed

Medium: Issue **SHOULD** be fixed

Low: Issue **MAY** be fixed

5. Security Assessment Summary

Overview

Repository: https://github.com/nqma-ime/alienx-contracts
Commit hash: 3969c403684173f7c2f6b1a133da3c33de26684b

Scope

The smart contracts included in the scope of the security review:

- src/AlienX.sol
- src/const/Constants.sol
- src/InfernoVault.sol
- src/utils/Math.sol
- src/libraries/UniswapV2Library.sol
- src/BuyAndBurn.sol
- src/Minting.sol
- src/BuyAndBurn/BaseBuyAndBurn.sol

Timeline

Audit start: October 13, 2024
Preliminary report: October 16, 2024

6. Executive Summary

Throughout the security review, conducted in the period October 13-16, 2024, 5 issues were discovered.

Findings count

High Risk	2
Medium Risk	2
Low Risk	1

7. Findings

7.1 High Findings

[H1] TitanX, sent to the LP Wallet, is not deducted from the distribution amount, causing failure in the distribution of TitanX

Severity: *High*

Context: Minting._distribute.sol#L236

Description: In the *_distribute* function of Minting.sol an initial amount of TitanX needs to be accumulated and sent to the LP Wallet before normal distributions can be initiated. The issue is that the TitanX tokens, dedicated to the LP Wallet, are not properly deducted from the entire distribution amount. As a result, TitanX that has already been sent to the LP Wallet will also be transferred to the various burning destinations, causing a revert, due to an insufficient token balance.

Recommendation: The TitanX tokens, sent to the LP Wallet, should be deducted from the *_amount* variable.

Resolution: Resolved.

[H2] Minting AlienX tokens will fail

Severity: *High*

Context: AlienX._update.sol#L77

Description: The AlienX token is intended to be a fee-on-transfer token, where a fee is deducted from the initiator of a transfer. However, a fee deduction will also be initiated whenever tokens are minted. In that scenario, the *ERC20._burn* function will be invoked with a *from*

parameter set to *address(0).* This will cause the minting of tokens to revert, leading to a complete DoS of the protocol.

Recommendation: Implement the fee-on-transfer logic only if the *from* and *to* parameters of *_update* are not *address(0).*

Resolution: Resolved.

7.2 Medium Findings

[M1] Insufficient slippage protection in BuyAndBurn1.sol

Severity: *Medium*

Context: BuyAndBurn1._swapTitanXForInferno.sol#L161

Description: In the constructor of the BuyAndBurn1 contract the *titanXToInfernoSlippage* variable is set to 80. It is later used in the *_swapTitanXForInferno* function to calculate the minimum number of tokens that should be received from the swap of TitanX tokens to Inferno tokens. The intended implementation should make sure that at least 80% of the expected Inferno tokens are received from the swap.
However, due to how the calculations are currently performed, the minimum number of received Inferno tokens can be as low as 20% of the expected value.
As a result, the *_swapTitanXForInferno* function becomes vulnerable to sandwich attacks, where the tokens received when swapping TitanX for Inferno will be much less than desired.

Recommendation: Set *titanXToInfernoSlippage* to 20 in the constructor of BuyAndBurn1.sol, to achieve the intended slippage protection.

Resolution: Resolved.

[M2] Interval distribution will be wrong in some cases

Severity: *Medium*

Context: BaseBuyAndBurn._calculateIntervals.sol#L193-L196

Description: In the *_calculcateIntervals* function of BaseBuyAndBurn.sol when multiple days have passed since the last burned interval start timestamp the number of TitanX tokens distributed for those days will be calculated incorrectly. This occurs due to an erroneous assumption that any days that are after the day of the last burned interval will also belong to different snapshots.
This is incorrect as each snapshot lasts for 7 days and it is highly likely that the day of the interval that is currently being observed is in the same snapshot as the day of the last burned interval.
Thus, the allocated TitanX tokens for such intervals will not be derived through the total TitanX distributed for their correct snapshot but will instead be a portion of the remaining token balance in the contract.

Recommendation: An interval's distributed TitanX tokens should be calculated as a portion of the current *totalTitanXDistributed* when the day of the interval belongs to the one-week period of the snapshot.

Resolution: Resolved.

7.3 Low Findings

[L1] Distribution of TitanX tokens may fail, due to rounding down to 0

Severity: Low

Context: Minting._distribute.sol#L249

Description: In the *_distribute* function of Minting.sol an initial amount of TitanX needs to be accumulated in order to fund the AlienX/TitanX LP, AlienX/Inferno LP and LP Wallet. After that, if there are any TitanX tokens left, normal distribution to a variety of burning destinations is initiated.
The TitanX tokens that are left are proportionally split into 9 destinations. Here is how the calculation for each burning destination is

performed: *_amount * destination_percent / 1e18.*
However, if the *_amount* is low enough the calculation will round down to zero, causing the function to revert when *distributeTitanXForBurning* is called.

Recommendation: Consider not making a call to *distributeTitanXForBurning* if the tokens allocated for the call are 0.

Resolution: Acknowledged.

7.4 Informational Findings

[I1] Account abstraction wallets cannot call *swapTitanXForInfernoAndSendToVault* as the *msg.sender* must be the same as the origin of the transaction (*tx.origin*). This incompatibility with AAW may prevent honest users from interacting with the protocol.