Titan X Security Review

Version 1.0

October 30, 2023

Conducted by:
Georgi Georgiev (Gogo), Independent Security Researcher

Table of Contents

1 About Gogo

Georgi Georgiev, known as Gogo, is an independent security researcher experienced in Solidity smart contract auditing and bug hunting. Having conducted over 40 solo and team smart contract security reviews, he consistently aims to provide top-quality security auditing services. He serves as a smart contract auditor at Paladin Blockchain Security, where he has been involved in security audits for notable clients such as LayerZero, TraderJoe, SmarDex, and other leading protocols.

For security consulting, you can contact him on Twitter, Telegram, or Discord - *@gogotheauditor*.

2 Disclaimer

Audits are a time, resource and expertise bound effort where trained experts evaluate smart contracts using a combination of automated and manual techniques to find as many vulnerabilities as possible. Audits can show the presence of vulnerabilities **but not their absence**.

3 Risk classification

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

3.1 Impact

- **High** - leads to a significant loss of assets in the protocol or significantly harms a group of users.
- **Medium** - only a small amount of funds can be lost or a functionality of the protocol is affected.
- **Low** - any kind of unexpected behaviour that's not so critical.

3.2 Likelihood

- **High** - direct attack vector; the cost is relatively low to the amount of funds that can be lost.
- **Medium** - only conditionally incentivized attack vector, but still relatively likely.
- **Low** - too many or too unlikely assumptions; provides little or no incentive.

3.3 Actions required by severity level

- **Critical** - client **must** fix the issue.
- **High** - client **must** fix the issue.
- **Medium** - client **should** fix the issue.
- **Low** - client **could** fix the issue.

4 Executive summary

Overview

Project Name	Titan X
Repository	https://github.com/jakesharpe777/ttx_private
Commit hash	af3d9e37520ec15600f589522bd20085c6a2b812
Resolution	a545172b7742f0dd0419f18bbb5b7c00bef29064
Repository	https://github.com/jakesharpe777/ttx_buyandburn_private
Commit hash	5985395c267ea8897087e680af0364a4bc0c9699
Resolution	6dcabb3568c67c1c47f6cc329b8832b65e0f40e9
Methods	Manual review & Fuzz testing

Scope

contracts/BurnInfo.sol
contracts/GlobalInfo.sol
contracts/MintInfo.sol
contracts/OwnerInfo.sol
contracts/StakeInfo.sol
contracts/TITANX.sol
contracts/BuyAndBurn.sol

Issues Found

Governance risk	1
Critical risk	0
High risk	0
Medium risk	1
Low risk	0

5 Findings

5.1 Governance risk

5.1.1 TitanX contract owner can burn UniswapV3 pair's liquidity and steal all WETH

Severity: *Governance risk*

Context: TITANX.sol#L324-L331, TITANX.sol#L356-L359

Description: The TITANX contract implements a functionality that allows the owner of the contract (the protocol admin address) to update the BuyAndBurn contract address. The purpose of this function is, as stated in a comment, "to change to new contract that supports UniswapV4+".

However, this introduces a centralization vulnerability due to the `burnLPTokens` function:

```
function burnLPTokens() external dailyUpdate {
    _burn(s_buyAndBurnAddress, balanceOf(s_buyAndBurnAddress));
}
```

In case the private key of the TITANX contract owner is leaked, an adversary can set the `s_buyAndBurnAddress` variable to any UniswapV3 pair contract address and then burn all TITANX tokens to steal the WETH liquidity from the corresponding pool.

Recommendation: It is recommended to use a strong and diverse multi-signature wallet contract with KYC-ed or doxx-ed participants as the TITANX contract owner, following best practices.

Resolution: Acknowledged. The TITANX contract is managed by an EOA at the time of completing this security review - 0x10129f3fe44dd32745d6e64e5d84cb84a524f114. Users should be aware of the aforementioned risk.

5.2 Medium risk

5.2.1 Burn approvals for mints and stakes may result in unexpected outcomes

Severity: *Medium risk*

Context: TITANX.sol#L965-L985, TITANX.sol#L649-L673

Description: Users are provided with the option to approve the burning of a portion of their mints and stakes by external protocols. They can specify the total amount of mints or stakes to be approved to specific addresses.

However, not all mint and stakes have the same properties, particularly in terms of the resulting number of TITANX tokens to be burned. If a user possesses both a mint for 1 million TITANX tokens and a mint for 100 million TITANX tokens, when they set the allowance for a particular project to 1 mint, they are technically approving both. This creates a potential issue where, in the event that the approved address is malicious, they can use the user's 100 million TITANX mint for burning, even if the user's intention was to burn only the smaller 1 million mint.

Recommendation: Consider allowing users to approve only specific mint and stake IDs. Alternatively, you can reuse the already implemented mechanism of setting an `amount` cap, but instead of reducing it by 1 for each mint or stake burned, reduce it by the corresponding amount of TITANX tokens burned.

Resolution: Acknowledged. Users will have to do their due diligence before they interact with burning protocols. Protocols can't do damage without user permission and the incentive is relatively low.