Helios Security Audit

Report Version 1.1

February 20, 2024

Conducted by:

George Hunter, Independent Security Researcher

Table of Contents

1 About George Hunter

George Hunter is a proficient and reputable independent smart contract security researcher with over 50 solo and team security engagements contributing to the security of numerous smart contract protocols in the past 2 years. Previously held roles include Lead Smart Contract Auditor at Paladin Blockchain Security and Smart Contract Engineer at Nexo. He has audited smart contracts for clients such as LayerZero, Euler, TraderJoe, Maverick, Ambire, and other leading protocols.

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 identify as many vulnerabilities as possible. Audits can reveal the presence of vulnerabilities **but cannot guarantee their absence**.

3 Risk classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	High	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** - involves a small loss of funds or affects a core functionality of the protocol.
- **Low** - encompasses any unexpected behavior that is non-critical.

3.2 Likelihood

- **High** - a direct attack vector; the cost is relatively low compared to the potential loss of funds.
- **Medium** - only a conditionally incentivized attack vector, with a moderate likelihood.
- **Low** - involves too many or unlikely assumptions; offers little to no incentive.

3.3 Actions required by severity level

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

4 Executive summary

Overview

Project Name	Helios
Repository	https://github.com/DudeGuy420/helios-contracts
Commit hash	e73ef3e07c58dbb88e5d47555605bfc0dd6ef13c
Mitigation review	95723f685c164f2e2bd3bad98bb2ba52bbc96caf
Methods	Manual review

Scope - Full review

contracts/Treasury.sol
contracts/HlxStake.sol

Scope - Changes review

contracts/BurnInfo.sol
contracts/GlobalInfo.sol
contracts/Helios.sol
contracts/MintInfo.sol
contracts/OwnerInfo.sol
contracts/StakeInfo.sol
libs/calcFunctions.sol
libs/constant.sol
libs/enum.sol
BuyAndBurn/BuyAndBurn.sol

Issues Found

High risk	2
Medium risk	1
Low risk	1

5 Findings

5.1 High

5.1.1 Wrong Uniswap pool used for obtaining ETH price

Severity: *High*

Context: BuyAndBurn.sol#L698

Description: In the previous version of BuyAndBurn, swaps were executed only from WETH to TitanX, but now swaps from TitanX to Helios are performed as well. For that reason, the getCurrentTitanPrice function was implemented which is used in the slippage checks:

```
function getCurrentTitanPrice() public view returns (uint256) {
    uint256 sqrtPriceX96 = getSqrtPriceX96(s_poolAddress);
    uint256 numerator1 = sqrtPriceX96 * sqrtPriceX96;
    uint256 numerator2 = 10 ** 18;
    uint256 price = FullMath.mulDiv(numerator1, numerator2, 1 << 192);
    price = TITANX < s_hlxAddress ? (1 ether * 1 ether) / price : price;

    return price;
}
```

It uses the s_poolAddress which is the TitanX:Helios UniswapV3 pool to fetch the sqrtPriceX96 from slot0 and convert to the actual token price.

The problem is that the s_poolAddress is used in the getCurrentEthPrice function as well which should actually use the TITANX_WETH_POOL:

```
function getCurrentEthPrice() public view returns (uint256) {
    uint256 sqrtPriceX96 = getSqrtPriceX96(s_poolAddress); // @audit wrong pool
    ...
}
```

Therefore, a TitanX:Helios price is returned when the expected is WETH:TitanX meaning that the WETH:TitanX transfers would either be blocked or the slippage protection will not work.

Recommendation: Use the TITANX_WETH_POOL constant instead of s_poolAddress in getCurrentEthPrice ().

Resolution: Resolved. The correct pool is now used.

5.1.2 Mint cost cap is reached sooner than intended

Severity: *High*

Context: GlobalInfo.sol#L171-L173

Description: The _dailyUpdate function performs the following check and update of the mint cost:

```
if (newMintCost > 1_000_000 ether) {
    newMintCost = CAPPED_MAX_MINT_COST;
}
```

The problem is that the CAPPED_MAX_MINT_COST constant has a value of 2_000_000_000e18 instead of 1_000_000. Therefore, the mint cost will reach its cap way sooner than intended.

Recommendation: Consider either using the CAPPED_MAX_MINT_COST constant in the if-check or changing the constant value from 2_000_000_000e18 to 1_000_000_000e18.

Resolution: Resolved. The correct constant is now used.

5.2 Medium

5.2.1 An adversary can block the burning functionality for stakers and minters

Severity: *Medium*

Context: Helios.sol#L143-L145

Description: The Helios contract implements the onBurn callback function called by the TitanX contract in the _burnAfter hook:

```
function onBurn(address, uint256 amount) external override {
    s_totalTitanBurned += amount;
}
```

The problem is that the function can be called by anyone. An adversary may exploit this by passing max uint256 as amount which would cause any sequential calls to the function by TitanX to revert effectively blocking the additional incentive given for users that burn TitanX tokens when starting mints and stakes.

Recommendation: Implement an access control modifier that allows only the TitanX contract to call the corresponding function.

Resolution: Resolved. The onBurn function now implements an access control check.

5.3 Low

5.3.1 Unsafe integer downcasting

Severity: *Low*

Context: GlobalInfo.sol, MintInfo.sol

Description: Multiple structs properties and storage variables of integer type are not using a full ethereum word (32 bytes) but smaller sizes. At the same time, many constant values have been increased which means that some variables might need more space in order not to overflow.

Recommendation: Consider removing all unsafe downcasts and using only full integers (uint256 and int256).

Resolution: Resolved. All relevant unsafe downcasts have been removed.