code4rena

XDEFI

Smart Contract Security Assessment

Version 1.0

Audit dates:	Aug 23 — Aug 26, 2024
Audited by:	SpicyMeatball 0x1771

Contents

1. Introduction

1.1 About Zenith

Zenith is an offering by Code4rena that provides consultative audits from the very best security researchers in the space. We focus on crafting a tailored security team specifically for the needs of your codebase.

Learn more about us at https://code4rena.com/zenith.

1.2 Disclaimer

This report reflects an analysis conducted within a defined scope and time frame, based on provided materials and documentation. It does not encompass all possible vulnerabilities and should not be considered exhaustive.

The review and accompanying report are presented on an "as-is" and "as-available" basis, without any express or implied warranties.

Furthermore, this report neither endorses any specific project or team nor assures the complete security of the project.

1.3 Risk Classification

SEVERITY LEVEL	IMPACT: HIGH	IMPACT: MEDIUM	IMPACT: LOW
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

2. Executive Summary

2.1 About XDEFI

One wallet for all your crypto Securely store, swap, and send Crypto and NFTs across 200+ blockchains.Connect to every dApp on Ethereum, Cosmos, BSC, Polygon, Solana, Bitcoin and more.

2.2 Scope

Repository	XDeFi-tech/xdefi-ctrl-migration
Commit Hash	5392346bc149c7ac509569ec8110a563846a92ea

2.3 Audit Timeline

DATE	EVENT
Aug 23, 2024	Audit start
Aug 26, 2024	Audit end
Nov 14, 2024	Report published

2.4 Issues Found

SEVERITY	COUNT
Critical Risk	0
High Risk	0
Medium Risk	1
Low Risk	0
Informational	1
Total Issues	2

3. Findings Summary

ID	DESCRIPTION	STATUS
M-1	Migration is vulnerable to permission frontrun	Resolved
I-1	Unused code in various contracts	Acknowledged

4. Findings

4.1 Medium Risk

A total of 1 medium risk findings were identified.

[M-1] Migration is vulnerable to permission frontrun

Severity: Medium	Status: Resolved

Context:

- XdefiToCtrlMigration.sol#L63
- XdefiToCtrlMigration.sol#L85
- XdefiToCtrlMigration.sol#L106
- XdefiToCtrlMigration.sol#L131

Description: Migration functions can be temporarily blocked with a permission frontrun. An attacker can call permit with the user's signature directly on the token contract and increment the nonce, reverting the migration tx:

```
       function permit(
           address owner_,
           address spender,
           uint256 value,
           uint256 deadline,
           uint8 v,
           bytes32 r,
           bytes32 s
       ) external override {
           require(owner_ != address(0), "ERC20: Owner cannot be 0");
           require(block.timestamp < deadline, "ERC20: Expired");
           bytes32 digest =
                           keccak256(
                   abi.encodePacked(
                       EIP191_PREFIX_FOR_EIP712_STRUCTURED_DATA,
                       DOMAIN_SEPARATOR,
 >>                    keccak256(abi.encode(PERMIT_SIGNATURE_HASH, owner_,
spender, value, nonces[owner_]++, deadline))
                   )
               );
           ---SNIP---
```

Recommendation: It is recommended to wrap `token.permit()` calls in a try-catch block to allow tx to continue if the permission has already been consumed:

```
function migrate(uint256 amount, uint256 deadline, uint8 v, bytes32 r,
bytes32 s) public {
+   try IERC20Permit(address(oldToken)).permit(msg.sender, address(this),
amount, deadline, v, r, s) {} catch {}
    // send tokens
}
```

XDEFI: The issue has been fixed with [PR-8]

Zenith: Verified.

4.2 Informational

A total of 1 informational findings were identified.

[I-1] Unused code in various contracts

Severity: Informational	Status: Acknowledged

Context:

- FixedToken.sol#L26
- ERC20.sol#L337

Description: `FixedToken.sol` has the `owner` argument in the `initToken` function that is never used:

```
    function initToken(string memory _name, string memory _symbol,
address _owner, uint256 _initialSupply) public  {
        _initERC20(_name, _symbol);
        _mint(msg.sender, _initialSupply);
    }
```

`ERC20,sol` has a function `_setupDecimals`, which is also not used:

```
    function _setupDecimals(uint8 decimals_) internal {
        _decimals = decimals_;
    }
```

Recommendation: Consider removing unused code and parameters from contracts.

XDEFI: Acknowledged.