SOLIDITY FINANCE

Pixelmon Staking

Smart Contract Audit Report

AUDIT SUMMARY

Pixelmon is building a new platform where users can stake NFTs and claim rewards.

For this audit, we reviewed the project team's PixelmonStaking contract at 0x217B53F8a059A5768aef26034b4A4ed2018d70f2 on the Ethereum Mainnet.

AUDIT FINDINGS

> *No findings were identified, though some centralized aspects are present.*
>
> *Date: September 14th, 2022.*
>
> *Updated: November 3rd, 2022 to reflect the project's Mainnet*

CONTRACT OVERVIEW

- *Users can stake Pixelmon NFTs into the contract by specifying each NFT's token ID and providing a valid signature.*
- *The returned signer address must have been added to the Signer role by the team in order for staking transactions to successfully occur.*
- *The user is sent 1 first-range Pixelmon Trainer NFT for each staked token.*
- *The team will set the Pixelmon and Pixelmon Trainer contracts upon deployment.*
- *Users cannot stake more than 25 NFTs per transaction.*
- *Each NFT can only ever be staked one time.*
- *The team must ensure that the contract has a sufficient balance of first-range Pixelmon Trainer NFTs to support stakes.*
- *Users can unstake any of their NFTs after the 21-day lock period has passed since staking.*
- *Users cannot unstake more than 25 NFTs per transaction.*
- *The owner can enable/disable Emergency mode at any time.*
- *When Emergency mode is enabled, users can unstake any of their NFTs at any time without enforcing the lock period restriction.*
- *The owner can add/remove addresses from the Signer role at any time.*
- *The owner can pause/unpause the contract at any time which disables all staking and unstaking.*

> - *The owner can withdraw any NFTs (besides Pixelmon NFTs) from the contract at any time.*
> - *The contract utilizes ReentrancyGuard to prevent reentrancy attacks in applicable functions.*

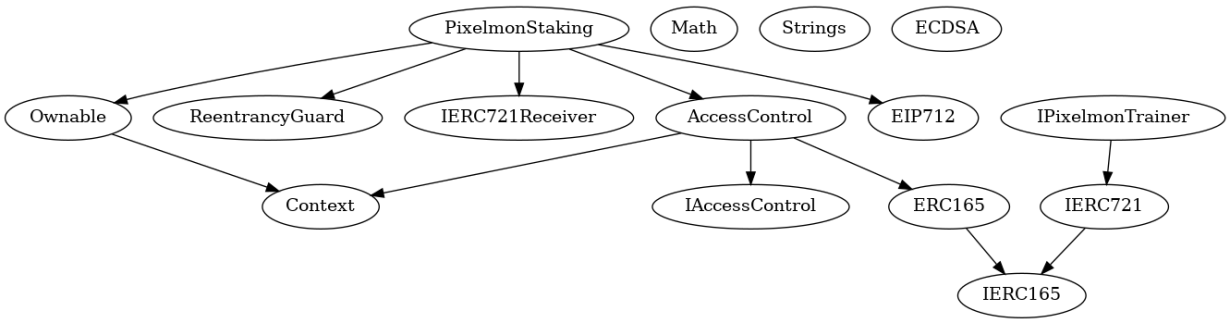
AUDIT RESULTS

Vulnerability Category	Notes	Result
Arbitrary Jump/Storage Write	N/A	PASS
Centralization of Control	The owner can pause all staking and unstaking at any time.	PASS
Compiler Issues	N/A	PASS
Delegate Call to Untrusted Contract	N/A	PASS
Dependence on Predictable	N/A	PASS

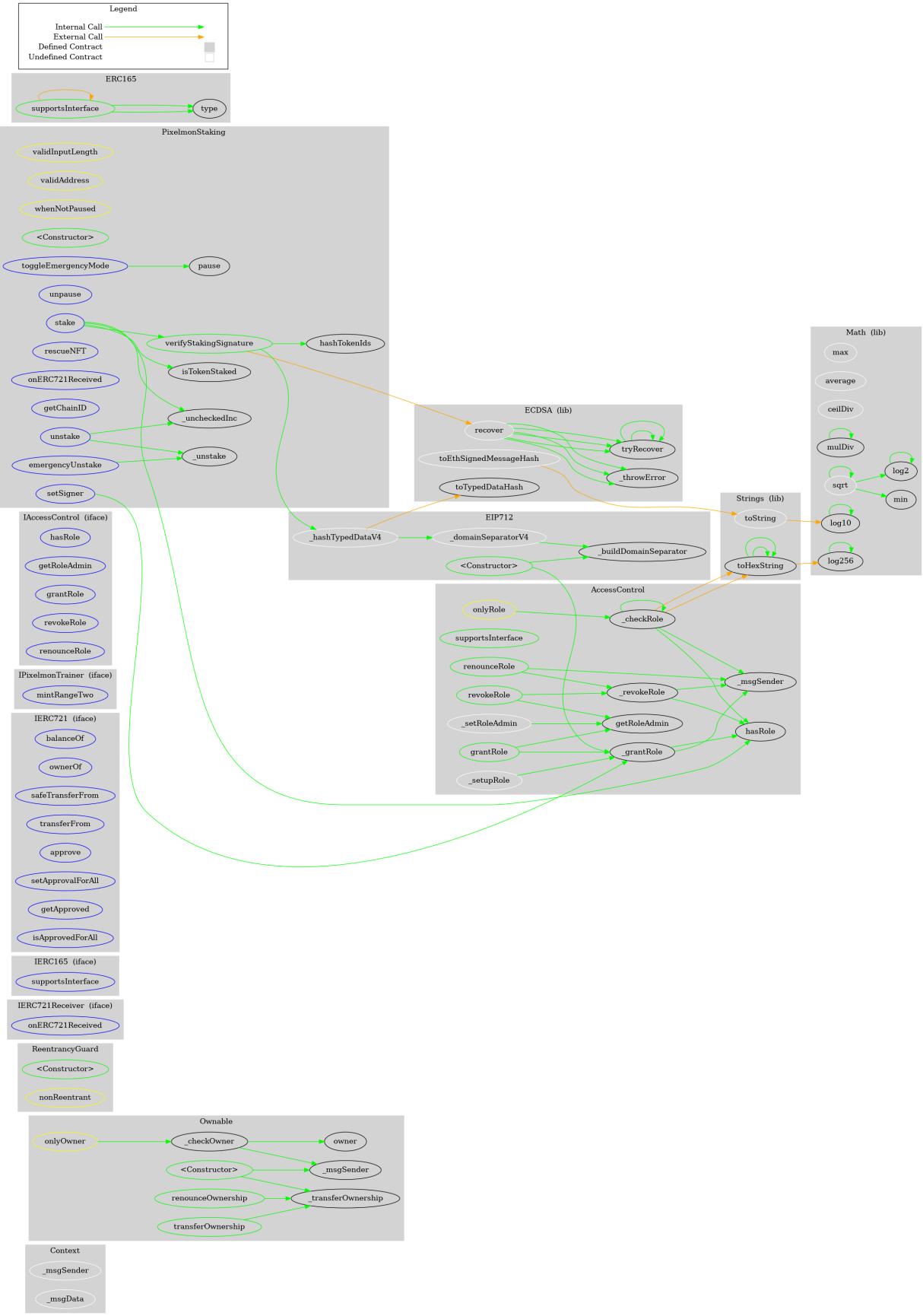
Vulnerability Category	Notes	Result
Ether/Token Theft	N/A	PASS
Flash Loans	N/A	PASS
Front Running	N/A	PASS
Improper Events	N/A	PASS
Improper Authorization Scheme	N/A	PASS
Integer Over/Underflow	N/A	PASS
Logical Issues	N/A	PASS
Oracle Issues	N/A	PASS
Outdated Compiler Version	N/A	PASS
Race Conditions	N/A	PASS

Vulnerability Category	Notes	Result
Signature Issues	N/A	PASS
Unbounded Loops	N/A	PASS
Unused Code	N/A	PASS
Overall Contract Safety		PASS

INHERITANCE CHART



FUNCTION GRAPH



FUNCTIONS OVERVIEW

```
($) = payable function
# = non-constant function

Int = Internal
Ext = External
Pub = Public

+  Context
   - [Int] _msgSender
   - [Int] _msgData

+  Ownable (Context)
   - [Pub]  #
   - [Pub] owner
   - [Int] _checkOwner
   - [Pub] renounceOwnership #
      - modifiers: onlyOwner
   - [Pub] transferOwnership #
      - modifiers: onlyOwner
   - [Int] _transferOwnership #

+  ReentrancyGuard
   - [Pub]  #

+ [Int] IERC721Receiver
   - [Ext] onERC721Received #

+ [Int] IERC165
   - [Ext] supportsInterface
```

```
    - [Ext] safeTransferFrom #
    - [Ext] safeTransferFrom #
    - [Ext] transferFrom #
    - [Ext] approve #
    - [Ext] setApprovalForAll #
    - [Ext] getApproved
    - [Ext] isApprovedForAll

+ [Int] IPixelmonTrainer (IERC721)
    - [Ext] mintRangeTwo #

+ [Int] IAccessControl
    - [Ext] hasRole
    - [Ext] getRoleAdmin
    - [Ext] grantRole #
    - [Ext] revokeRole #
    - [Ext] renounceRole #

+ [Lib] Math
    - [Int] max
    - [Int] min
    - [Int] average
    - [Int] ceilDiv
    - [Int] mulDiv
    - [Int] mulDiv
    - [Int] sqrt
    - [Int] sqrt
    - [Int] log2
    - [Int] log2
    - [Int] log10
    - [Int] log10
    - [Int] log256
```

```
+ [Lib] Strings
   - [Int] toString
   - [Int] toHexString
   - [Int] toHexString
   - [Int] toHexString

+  ERC165 (IERC165)
   - [Pub] supportsInterface

+  AccessControl (Context, IAccessControl, ERC165)
   - [Pub] supportsInterface
   - [Pub] hasRole
   - [Int] _checkRole
   - [Int] _checkRole
   - [Pub] getRoleAdmin
   - [Pub] grantRole #
      - modifiers: onlyRole
   - [Pub] revokeRole #
      - modifiers: onlyRole
   - [Pub] renounceRole #
   - [Int] _setupRole #
   - [Int] _setRoleAdmin #
   - [Int] _grantRole #
   - [Int] _revokeRole #

+ [Lib] ECDSA
   - [Prv] _throwError
   - [Int] tryRecover
   - [Int] recover
   - [Int] tryRecover
   - [Int] recover
   - [Int] tryRecover
```

```
    - [Int] toEthSignedMessageHash
    - [Int] toTypedDataHash

+  EIP712
    - [Pub]  #
    - [Int] _domainSeparatorV4
    - [Prv] _buildDomainSeparator
    - [Int] _hashTypedDataV4

+  PixelmonStaking (IERC721Receiver, Ownable, Reentranc
    - [Pub]  #
       - modifiers: EIP712
    - [Ext] setSigner #
       - modifiers: onlyOwner
    - [Pub] pause #
       - modifiers: onlyOwner
    - [Ext] unpause #
       - modifiers: onlyOwner
    - [Ext] toggleEmergencyMode #
       - modifiers: onlyOwner
    - [Ext] rescueNFT #
       - modifiers: onlyOwner,nonReentrant,validAddress,
    - [Ext] onERC721Received
    - [Pub] hashTokenIds
    - [Pub] verifyStakingSignature
    - [Ext] getChainID
    - [Ext] stake #
       - modifiers: nonReentrant,whenNotPaused,validInpu
    - [Ext] unstake #
       - modifiers: nonReentrant,whenNotPaused,validInpu
    - [Ext] emergencyUnstake #
       - modifiers: nonReentrant
```

```
    - [Prv] _unstake #
    - [Prv] _uncheckedInc
```

ABOUT SOLIDITY FINANCE

Solidity Finance was founded in 2020 and quickly grew to have one of the most experienced and well-equipped smart contract auditing teams in the industry. Our team has conducted 1300+ solidity smart contract audits covering all major project types and protocols, securing a total of over $10 billion U.S. dollars in on-chain value.
Our firm is well-reputed in the community and is trusted as a top smart contract auditing company for the review of solidity code, no matter how complex. Our team of experienced solidity smart contract auditors performs audits for tokens, NFTs, crowdsales, marketplaces, gambling games, financial protocols, and more!

Contact us today to get a free quote for a smart contract audit of your project!

WHAT IS A SOLIDITY AUDIT?

Typically, a smart contract audit is a comprehensive review process designed to discover logical errors, security vulnerabilities, and optimization opportunities within code. A *Solidity Audit* takes this a step further by verifying economic logic to ensure the stability of smart contracts and highlighting privileged functionality to create a report that is easy to understand for developers and community members alike.

HOW DO I INTERPRET THE FINDINGS?

Each of our Findings will be labeled with a Severity level. We always recommend the team resolve High, Medium, and Low severity findings prior to deploying the code to the mainnet. Here is a breakdown on what each Severity level means for the project:

- **High** severity indicates that the issue puts a large number of users' funds at risk and has a high probability of exploitation, or the smart contract contains serious logical

- **Medium** severity issues are those which place at least some users' funds at risk and has a medium to high probability of exploitation.
- **Low** severity issues have a relatively minor risk association; these issues have a low probability of occurring or may have a minimal impact.
- **Informational** issues pose no immediate risk, but inform the project team of opportunities for gas optimizations and following smart contract security best practices.

GO HOME