SAFUAUDIT

SMART CONTRACT SECURITY ASSESSMENT

PROJECT:

SWDESIGNMETAVERSITY

DATE:

01 MARCH, 2023

✉ https://t.me/SafuAudit

🌐 www.safuaudit.com

Introduction

Client	SWDesignMetaversity
Language	Solidity
Contract Address	0xc99bccc39a2da1ad99987e9e174c213dcc968c23
Owner	0x25AF8949F4fCB79476e4F7a3e7a28cA9F2F5a7b6
Deployer	0x25AF8949F4fCB79476e4F7a3e7a28cA9F2F5a7b6
SHA-256 Hash	a02efcac860a4b016e5025aba7062d4445fdf6eb
Decimals	N/A
Supply	10000
Platform	Ethereum
Compiler	v0.8.19+commit.7dd6d404
Optimization	No with 200 runs
Website	https://swdesignmetaversity.io/
Twitter	https://twitter.com/swdmnft
Telegram	https://t.me/swdmetaversity

Overview

Fees

- Buy fees: 0%
- Sell fees: 5%

Fees privileges

- The owner can change the amount of royalties

Ownership

- Owned

Minting

- N/A

Max Tx Amount

- N/A

Pause

- Can't pause

Blacklist

- Can't blacklist

Other Privileges

- N/A

Table Of Contents

01 Intro

Introduction

Overview

Risk classification

02 Contract inspection

Contract Inspection

Inheritance Tree

04 Findings

Vulnerabilities Test

Findings list

Issues description

05 Conclusions

Disclaimer

Rating

Conclusion

Risk Classification

Critical

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium

Issues on this level could potentially bring problems and should eventually be fixed.

Minor

Issues on this level are minor details and warning that can remain unfixed but would be better fixed at some point in the future

Informational

Information level is to offer suggestions for improvement of efficacity or security for features with a risk free factor.

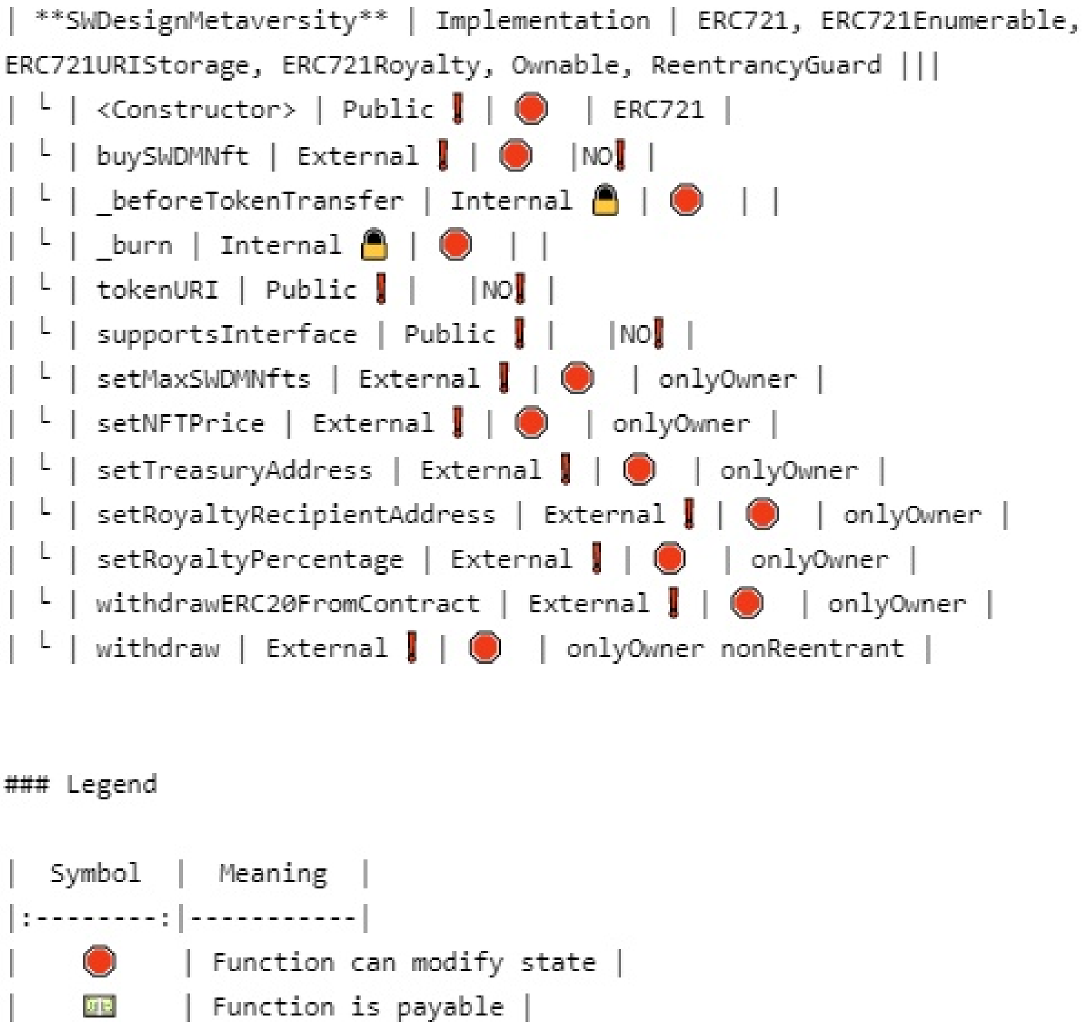
Contract Inspection

```
|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| SWDesignMetaversity.sol | a02efcac860a4b016e5025aba7062d4445fdf6eb |


### Contracts Description Table

|  Contract  |         Type        |       Bases      |                  |
|:------:|:-------------------:|:------------:|:--------------:|:------------|
|    └      |  **Function Name**  |  **Visibility**  |  **Mutability**  |  **Modifiers**
||||||
| **IERC20** | Interface |  |||
| **ReentrancyGuard** | Implementation |  |||
| **Math** | Library |  |||
| **Strings** | Library |  |||
| **Context** | Implementation |  |||
| **Ownable** | Implementation | Context |||
| **Address** | Library |  |||
| **IERC721Receiver** | Interface |  |||
| **IERC165** | Interface |  |||
| **IERC2981** | Interface | IERC165 |||
| **ERC165** | Implementation | IERC165 |||
| **ERC2981** | Implementation | IERC2981, ERC165 |||
| **IERC721** | Interface | IERC165 |||
| **IERC721Enumerable** | Interface | IERC721 |||
| **IERC721Metadata** | Interface | IERC721 |||
| **ERC721** | Implementation | Context, ERC165, IERC721, IERC721Metadata |||
| **ERC721Royalty** | Implementation | ERC2981, ERC721 |||
| **ERC721URIStorage** | Implementation | ERC721 |||
| **ERC721Enumerable** | Implementation | ERC721, IERC721Enumerable |||
```

Contract Inspection

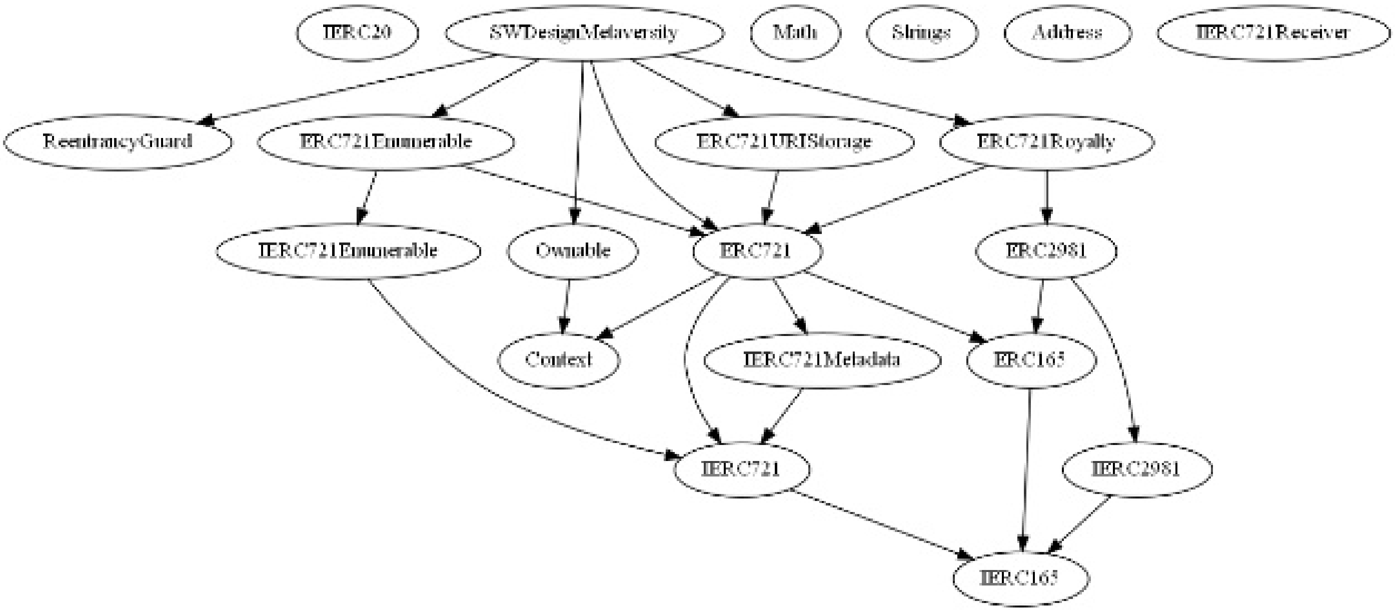


```
| **SWDesignMetaversity** | Implementation | ERC721, ERC721Enumerable,
ERC721URIStorage, ERC721Royalty, Ownable, ReentrancyGuard |||
| └ | <Constructor> | Public ❗ | 🛑  | ERC721 |
| └ | buySWDMNft | External ❗ | 🛑  |NO❗ |
| └ | _beforeTokenTransfer | Internal 🔒 | 🛑  | |
| └ | _burn | Internal 🔒 | 🛑  | |
| └ | tokenURI | Public ❗ |   |NO❗ |
| └ | supportsInterface | Public ❗ |   |NO❗ |
| └ | setMaxSWDMNfts | External ❗ | 🛑  | onlyOwner |
| └ | setNFTPrice | External ❗ | 🛑  | onlyOwner |
| └ | setTreasuryAddress | External ❗ | 🛑  | onlyOwner |
| └ | setRoyaltyRecipientAddress | External ❗ | 🛑  | onlyOwner |
| └ | setRoyaltyPercentage | External ❗ | 🛑  | onlyOwner |
| └ | withdrawERC20FromContract | External ❗ | 🛑  | onlyOwner |
| └ | withdraw | External ❗ | 🛑  | onlyOwner nonReentrant |


### Legend

|  Symbol  |  Meaning  |
|:--------:|-----------|
|    🛑    | Function can modify state |
|    💵    | Function is payable |
```

Contract Inheritance



Inheritance is a feature of the object-oriented programming language. It is a way of extending the functionality of a program, used to separate the code, reduces the dependency, and increases the re-usability of the existing code. Solidity supports inheritance between smart contracts, where multiple contracts can be inherited into a single contract.

Vulnerabilities Test

Test Name	Result
Function Default Visibility	Passed
Integer Overflow and Underflow	Passed
Outdated Compiler Version	Passed
Floating Pragma	Passed
Unchecked Call Return Value	Passed
Unprotected Ether Withdrawal	Passed
Unprotected SELF-DESTRUCT Instruction	Passed
Reentrancy	Passed
State Variable Default Visibility	Passed
Uninitialized Storage Pointer	Passed
Assert Violation	Passed
Use of Deprecated Solidity Functions	Passed
Delegate Call to Untrusted Callee	Passed
DoS with Failed Call	Passed
Transaction Order Dependence	Passed
Authorization through tx.origin	Passed
Block values as a proxy for time	Passed
Signature Malleability	Passed
Incorrect Constructor Name	Passed

Vulnerabilities Test

Test Name	Result
Shadowing State Variables	Passed
Weak Sources of Randomness from Chain Attributes	Passed
Missing Protection against Signature Replay Attacks	Passed
Lack of Proper Signature Verification	Passed
Requirement Violation	Passed
Write to Arbitrary Storage Location	Passed
Incorrect Inheritance Order	Passed
Insufficient Gas Griefing	Passed
Arbitrary Jump with Function Type Variable	Passed
DoS With Block Gas Limit	Passed
Typographical Error	Passed
Right-To-Left-Override control character (U+202E)	Passed
Presence of unused variables	Passed
Unexpected Ether balance	Passed
Hash Collisions With Multiple Variable Length Arguments	Passed
Message call with the hardcoded gas amount	Passed
Code With No Effects	Passed
Unencrypted Private Data On-Chain	Passed

Findings

ID	Category	Issue	Severity
CS-01	Coding Standards	Multiple Pragma Calls With Same Version	Optimization
CE-OF	Centralization	Owner Accessible Functions	Minor

CS-01 Multiple Pragma Calls With Same Version

Lines # multiple lines

```
pragma solidity ^0.8.19;
```

Description

Within the SWDesignMetaversity file, there are multiple contracts and in turn are multiple declarations of the solidity version to use.

Recommendation

Although this does not affect the operation of the contract, it is recommended to unify all these compiler declarations in a single declaration at the beginning of the file.

CE-OF Owner Accessible Functions

Lines # multiple lines

```
| └ | setMaxSWDMNfts
| └ | setNFTPrice
| └ | setTreasuryAddress
| └ | setRoyaltyRecipientAddress
| └ | setRoyaltyPercentage
| └ | withdrawERC20FromContract
| └ | withdraw
```

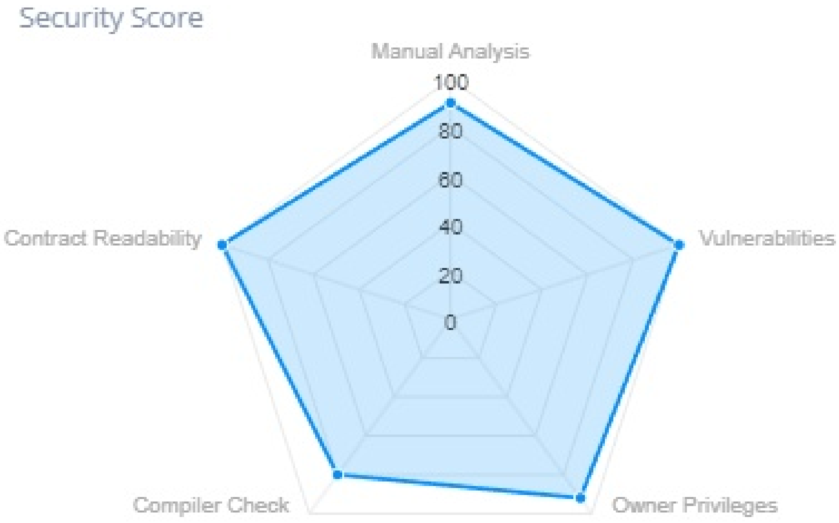
Description

The role OnlyOwner has authority over the above functions that can manipulate the project functionality. Any compromise to the owner account may allow a hacker to take advantage of this authority.

Recommendation

We advise the client to carefully manage the privilege accounts' private key to avoid any potential risks of being hacked.

Security Score



Manual Analysis Score	90
Vulnerabilities Score	100
Contract Readability Score	92
Owner Privileges	80
Compiler Score	100
Total	**92.4**

Conclusion

SWDesignMetaversityNFT Smart Contract uses ERC721 contract, designed for the launch of an NFT collection with integrated purchase function. These NFTs are bought using the SWDM ERC20 token as a payment method.

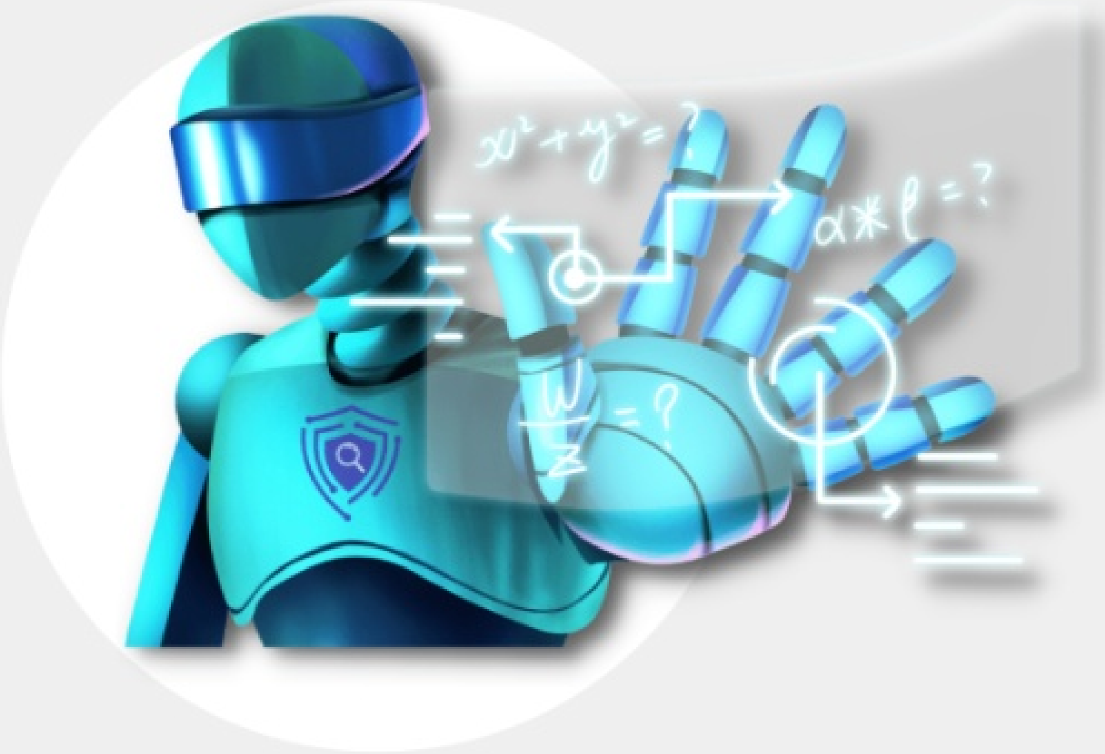
Disclaimer

SafuAudit.com is not a financial institution and the information provided on this website does not constitute investment advice, financial advice, trading advice, or any other sort of advice. You should not treat any of the website's content as such. Investing in crypto assets carries a high level of risk and does not hold guarantees for not sustaining financial loss due to their volatility.

Accuracy of Information
SafuAudit will strive to ensure the accuracy of the information listed on this website although it will not hold any responsibility for any missing or wrong information. SafuAudit provides all information as is. You understand that you are using any and all information available here at your own risk. Any use or reliance on our content and services is solely at your own risk and discretion.

The purpose of the audit is to analyze the on-chain smart contract source code and to provide a basic overview of the project.

While we have used all the information available to us for this straightforward investigation, you should not rely on this report only — we recommend proceeding with several independent audits Be aware that smart contracts deployed on a blockchain aren't secured enough against external vulnerability or a hack. Be aware that active smart contract owner privileges constitute an elevated impact on the smart contract safety and security. Therefore, SafuAudit does not guarantee the explicit security of the audited smart contract. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.



"Only in growth, reform, and change, paradoxically enough, is true security to be found."