

Jul 15, 2024



Security Assessment

MintSwap404NFT

Professional Service

Table of Contents

1. Overview

- 1.1. [Executive Summary](#)
- 1.2. [Project Summary](#)
- 1.3. [Assessment Summary](#)
- 1.4. [Assessment Scope](#)

2. Checklist

3. Findings

- [H-01: Return Value of 0 From ecrecover Is Not Checked](#)
- [H-02: Missing the Returning of the Residual msg.value](#)
- [M-03: Centralization Risk in Off-Chain Benefit Assessments](#)
- [I-04: Use the Checks-Effects-Interactions Pattern to Update the State](#)
- [I-05: No Check of Address Params with Zero Address](#)
- [I-06: Function Visibility Can Be External](#)
- [I-07: Parameters Should Be Declared as Calldata](#)
- [I-08: Floating Pragma](#)

4. Disclaimer

5. Appendix

1. Overview

1.1. Executive Summary

MintSwap404NFT is a project about ERC-404 tokens. Users can earn ETH rewards by staking specific ERC-404 tokens, and receive ERC-404 token rewards through liquidity mining. This report has been prepared for NFTScan to discover issues and vulnerabilities in the source code of this project as well as any contract dependencies that were not part of an officially recognized library.

Conducted by Static Analysis, Formal Verification and Manual Review, we have identified 2 High, 1 Medium and 5 Informational issues in commit 078b308.

The project team resolved security vulnerability described in H-01 and all the informational issues in commit 1df4c9e. About issues described in H-02 and M-03, the project team has confirmed that the implementation in commit 078b308 complies with the project design and decided to keep no change.

1.2. Project Summary

Project Name	MintSwap404NFT
Platform	Mint
Language	Solidity
Codebase	<p>Audit 1:</p> <ul style="list-style-type: none">https://github.com/MintSwapFinance/Contract-MintSwap404NFT/tree/078b308234acc26dfa4e2e57a1788d9c3fc5b593 <p>Final Audit:</p> <ul style="list-style-type: none">https://github.com/MintSwapFinance/Contract-MintSwap404NFT/tree/1df4c9e70ff8b89b8b390b7680468f84e151896c

1.3. Assessment Summary

Delivery Date	Jul 15, 2024
Audit Methodology	Static Analysis, Formal Verification, Manual Review

1.4. Assessment Scope

ID	File	File Hash
1	/src/lib/DoubleEndedQueue.sol	55cf2d2aff9dab91d1a82002638e69c4
2	/src/erc404/MintSwap404NFT.sol	d9a30f81a528cf875684744e94a8d208
3	/src/rewards/MintSwap404NFTRewards.sol	4f0b9273d81715c942c9c78f6d9f0588

ID	File	File Hash
4	/src/metadata/MetadataRenderer.sol	3d95845e45ebc66f619770c481ef558d
5	/src/erc404/IERC404.sol	f4d24eb72af5902eaf34818ef1a1a46d
6	/src/metadata/IMetadataRenderer.sol	f803ae26c529388183633636c58fd6ec
7	/src/erc404/ERC404.sol	7ecf9031109c3b4710608194af50e00b
8	/src/lib/ERC20Events.sol	4f7c1dae60d959fdf6bd8d803a84ddaa6
9	/src/stake/MintSwap404NFTStake.sol	535185d841e46dfc188e8513ad29c11f
10	/src/lib/ERC721Events.sol	c466d406bc52c2f09e1de1c8f17b1d13

2. Checklist

2.1. Code Security

Reentrancy	DelegateCall	Integer Overflow
Input Validation	Unchecked this.call	Frozen Money
Arbitrary External Call	Unchecked Owner Transfer	Do-while Continue
Right-To-Left-Override Character	Unauthenticated Storage Access	Risk For Weak Randomness
TxOrigin	Missing Checks for Return Values	Diamond Inheritance
ThisBalance	VarType Deduction	Array Length Manipulation
Uninitialized Variable	Shadow Variable	Divide Before Multiply
Affected by Compiler Bug		

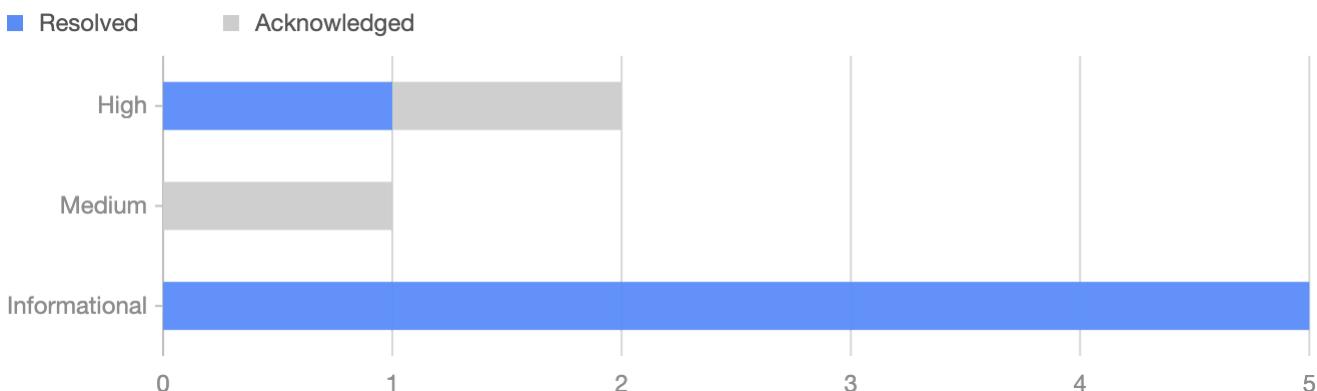
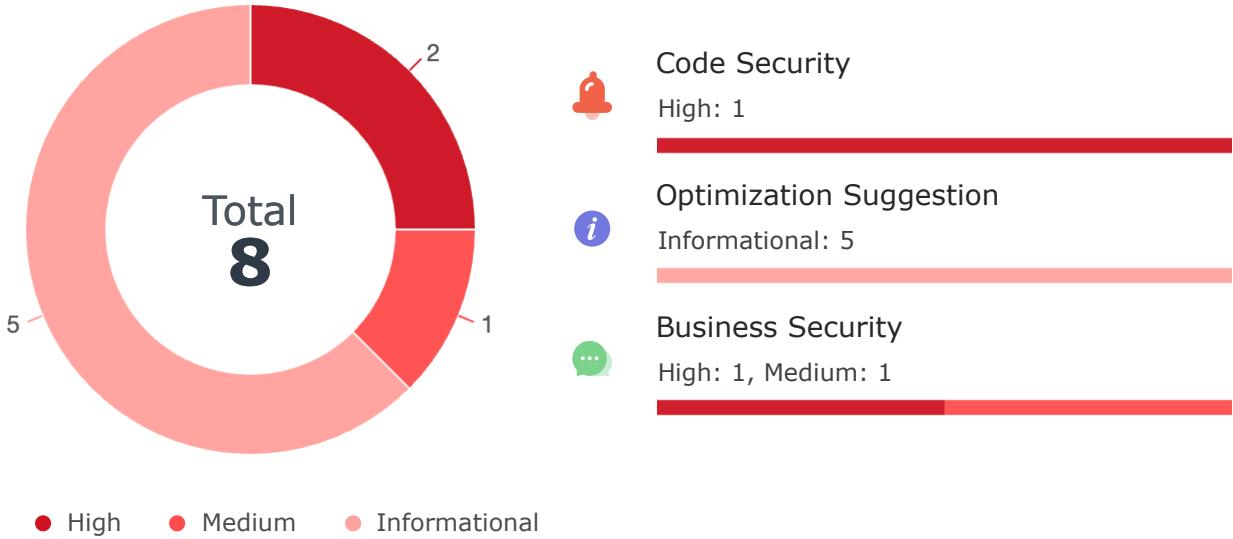
2.2. Optimization Suggestion

Compiler Version	Improper State Variable Modification
Function Visibility	Deprecated Function
Externally Controlled Variables	Code Style
Constant Specific	Event Specific
Return Value Unspecified	Inexistent Error Message
State Variable Defined Without Storage Location	Import Issue
Compare With Timestamp/Block Number/Blockhash	Constructor in Base Contract Not Implemented
Delete Struct Containing the Mapping Type	Usage of '=-'
Paths in the Modifier Not End with "_" or Revert	Non-payable Public Functions Use msg.value
Lack of SafeMath	Compiler Error/Warning
Tautology Issue	Loop Depends on Array Length
Redundant/Duplicated/Dead Code	Code Complexity/Code Inefficiency
Undeclared Resource	Optimizable Return Statement
Unused Resource	

2.3. Business Security

The Code Implementation is Consistent With Comments, Project White Papers and Other Materials
Permission Check
Address Check

3. Findings



ID	Title	Category	Severity	Status
H-01	Return Value of 0 From ecrecover Is Not Checked	Code Security	● High	Resolved
H-02	Missing the Returning of the Residual msg.value	Business Security	● High	Acknowledged
M-03	Centralization Risk in Off-Chain Benefit Assessments	Business Security	● Medium	Acknowledged
I-04	Use the Checks-Effects-Interactions Pattern to Update the State	Optimization Suggestion	● Informational	Resolved
I-05	No Check of Address Params with Zero Address	Optimization Suggestion	● Informational	Resolved
I-06	Function Visibility Can Be External	Optimization Suggestion	● Informational	Resolved
I-07	Parameters Should Be Declared as Calldata	Optimization Suggestion	● Informational	Resolved
I-08	FloatingPragma	Optimization Suggestion	● Informational	Resolved

H-01: Return Value of 0 From ecrecover Is Not Checked

High: Code Security



File Location:

/src/rewards/MintSwap404NFTRewards.sol:71
/src/stake/MintSwap404NFTStake.sol:108

Description

The `_verifySigner` function is designed to authenticate the legitimacy of user-provided signatures, thereby enabling the withdrawal of mining and staking rewards. It utilizes the built-in `ecrecover` function to recover the address that signed the message.

However, should the signature be invalid, `ecrecover` merely returns a zero address, rather than reverting (as detailed in the *Solidity* documentation

<https://docs.soliditylang.org/en/latest/units-and-global-variables.html#mathematical-and-cryptographic-functions>).

Consequently, if the signer address (the state variable `signer`) is either unset or erroneously set to zero, any arbitrary signature could pass this validity check, thus posing a significant security risk where all rewards could potentially be stolen.

/src/rewards/MintSwap404NFTRewards.sol

```
69         uint8 _v
70     ) internal view returns (address _signer) {
71     _signer = ecrecover(
72         keccak256(
73             abi.encodePacked(
```

/src/stake/MintSwap404NFTStake.sol

```
106        uint8 _v
107    ) internal view returns (address _signer) {
108    _signer = ecrecover(
109        keccak256(
110            abi.encodePacked(
```

Recommendation

Ensure that the return value of `ecrecover` is not zero to confirm the validity of the signature. Alternatively, consider utilizing the `ECDSA` contract from *OpenZeppelin* (<https://docs.openzeppelin.com/contracts/5.x/api/utils#ECDSA>) to more securely recover the address that signed the message.

Alleviation

Resolved in commit 1df4c9e. The project team addressed the issue by utilizing the `ECDSA` library to recover the address that signed the message.

H-02: Missing the Returning of the Residual msg.value



High: Business Security

File Location: /src/erc404/MintSwap404NFT.sol:106

Description

The `publicSale` function facilitates the public sale of the ERC-404 tokens. However, it currently lacks the capability to refund any excess Ether sent by buyers beyond the token's selling price, thereby rendering the residual `msg.value` to be lost.

/src/erc404/MintSwap404NFT.sol

```
106  function publicSale(uint number0fTokens) external payable
107    isPublicSaleTime {
108      require(number0fTokens > 0 && _publicMintedCount + number0fTokens <=
109          SALE_TOTAL_COUNT, "Mint number0fTokens exceeds limit");
110      require(PUBLIC_SALE_PRICE * number0fTokens <= msg.value, "Not Enough
111          ETH value to mint tokens");
112    }
```

Recommendation

It is recommended to incorporate logic within the `publicSale` function to facilitate the return of any residual `msg.value` to the buyer, ensuring equitable transaction processing.

Alleviation

The project team has confirmed that this implementation complies with the project design and decided to keep no change.

M-03: Centralization Risk in Off-Chain Benefit Assessments

Medium: Business Security



File Location:

/src/rewards/MintSwap404NFTRewards.sol:34
/src/stake/MintSwap404NFTStake.sol:73

Description

The functions `withdrawBenefits` in the `MintSwap404NFTRewards` contract and `withdrawStakeBenefits` in the `MintSwap404NFTStake` contract enable users to withdraw their mining and staking rewards with messages signed by the protocol.

Yet, the computation of these rewards occurs entirely off-chain, presenting a significant centralization threat to the protocol. Should the off-chain service calculating these benefits contains bugs or be compromised, it could break the entire integrity of the protocol.

/src/rewards/MintSwap404NFTRewards.sol

```
34  function withdrawBenefits(
35      uint256 _amount,
36      bytes32 _r,
37      bytes32 _s,
38      uint8 _v
39  ) external {
40      address sender = msg.sender;
41      require(_verifySigner(sender, _amount, _r, _s, _v) == signer,
42              "Invalid signer");
43      require(_amount > alreadyClaim[sender], "Invalid withdraw amount");
44
45      uint256 canClaimAmount = _amount - alreadyClaim[sender];
46      IERC404(mintswap404NFT).transferFrom(rewardsAccount, sender,
47          canClaimAmount);
48      alreadyClaim[sender] = _amount;
49      emit WithdrawRewardsBenefits(sender, canClaimAmount);
50  }
```

/src/stake/MintSwap404NFTStake.sol

```
73  function withdrawStakeBenefits(
74      uint256 _amount,
75      bytes32 _r,
76      bytes32 _s,
77      uint8 _v
78  ) external nonReentrant {
79      address payable sender = payable(msg.sender);
80      require(_verifySigner(sender, _amount, _r, _s, _v) == signer,
81              "Invalid signer");
82      require(_amount > alreadyClaim[sender], "Invalid withdraw amount");
```

```
83     uint256 canClaimAmount = _amount - alreadyClaim[sender];
84     (bool success, ) = sender.call{value: canClaimAmount}(new bytes(0));
85     require(success, 'ETH transfer failed');
86
87     alreadyClaim[sender] = _amount;
88     emit WithdrawStakeBenefits(sender, canClaimAmount);
89 }
```

Recommendation

Consider implementing the benefit calculation on-chain to enhance security and transparency.

Alleviation

The project team acknowledged the issue but has decided not to make any changes due to the complexity of the reward distribution logic.

I-04: Use the Checks-Effects-Interactions Pattern to Update the State

Informational: Optimization Suggestion



File Location:

/src/erc404/MintSwap404NFT.sol:100,101,102,103,110,111,118,119
/src/rewards/MintSwap404NFTRewards.sol:45,46
/src/stake/MintSwap404NFTStake.sol:63,64

Description

To enhance the security of the code, it is recommended to use the Checks-Effects-Interactions pattern to update the state. For example, in the `withdraw` function of the `MintSwap404NFTStake` contract, it is suggested to first delete `stakedTokens[tokenIds[i]]` and then transfer the ERC-404 NFT to the caller, in order to better comply with code safety standards and enhance code security.

/src/erc404/MintSwap404NFT.sol

```
91  function wlSale() external payable isWLSaleTime {
92      address account = _msgSender();
93      if (!whitelist[account]) revert UnauthorizedMinter(account);
94
95      require(!wlMinted[account], "This account has already WL minted");
96      require(_wlMintedCount + 1 <= WL_SALE_COUNT, "WL mint exceeds
97          limit");
98      require(_publicMintedCount + 1 <= SALE_TOTAL_COUNT, "Mint exceeds
99          limit");
100     require(WHITELIST_SALE_PRICE <= msg.value, "Not Enough ETH value to
101         WL mint tokens");
102
103     _mintERC20(account, units);
104     wlMinted[account] = true;
105     _publicMintedCount++;
106     _wlMintedCount++;
107 }
108
109 function publicSale(uint number0fTokens) external payable
110     isPublicSaleTime {
111     require(number0fTokens > 0 && _publicMintedCount + number0fTokens <=
112         SALE_TOTAL_COUNT, "Mint number0fTokens exceeds limit");
113     require(PUBLIC_SALE_PRICE * number0fTokens <= msg.value, "Not Enough
114         ETH value to mint tokens");
115
116     _mintERC20(_msgSender(), number0fTokens * units);
117     _publicMintedCount += number0fTokens;
118 }
119
120 function mintRewards(address exemptAddress, uint256 amount) external
121     onlyOwner {
122     require(erc721TransferExempt(exemptAddress), "The address is not
123         erc721TransferExempt");
```

```

116     require(amount > 0 && _mintswapMintedCount + amount <=
117         MINTSWAP_REWARDS_COUNT, "The maximum mint rewards quantity cannot
118         exceed 7000");
119     _mintERC20(exemptAddress, amount * units);
120     _mintswapMintedCount += amount;
121 }
```

/src/rewards/MintSwap404NFTRewards.sol

```

34     function withdrawBenefits(
35         uint256 _amount,
36         bytes32 _r,
37         bytes32 _s,
38         uint8 _v
39     ) external {
40         address sender = msg.sender;
41         require(_verfySigner(sender, _amount, _r, _s, _v) == signer,
42             "Invalid signer");
43         require(_amount > alreadyClaim[sender], "Invalid withdraw amount");
44
45         uint256 canClaimAmount = _amount - alreadyClaim[sender];
46         IERC404(mintswap404NFT).transferFrom(rewardsAccount, sender,
47             canClaimAmount);
48         alreadyClaim[sender] = _amount;
49         emit WithdrawRewardsBenefits(sender, canClaimAmount);
50     }
```

/src/stake/MintSwap404NFTStake.sol

```

56     function withdraw(uint256[] calldata tokenIds) external {
57         require(tokenIds.length > 0, "Withdrawing zero tokens");
58         address sender = msg.sender;
59
60         for (uint256 i = 0; i < tokenIds.length; ) {
61             address nftOwner = stakedTokens[tokenIds[i]];
62             require(sender == nftOwner, "Invalid tokenId");
63             IERC404(mintswap404NFT).transferFrom(address(this), sender,
64                 tokenIds[i]);
65             delete stakedTokens[tokenIds[i]];
66             unchecked {
67                 ++i;
68             }
69         }
70         emit TokensWithdraw(sender, tokenIds);
71     }
```

Recommendation

Suggested modifications are as follows:

/src/erc404/MintSwap404NFT.sol:

```

function wlSale() external payable isWLSaleTime {
    address account = _msgSender();
    if (!whitelist[account]) revert UnauthorizedMinter(account);

    require(!wlMinted[account], "This account has already WL minted");
    require(_wlMintedCount + 1 <= WL_SALE_COUNT, "WL mint exceeds limit");
    require(WHITELIST_SALE_PRICE <= msg.value, "Not Enough ETH value to WL
mint tokens");

    wlMinted[account] = true;
    _publicMintedCount++;
    _wlMintedCount++;
    _mintERC20(account, units);
}

function publicSale(uint number0fTokens) external payable isPublicSaleTime {
    require(number0fTokens > 0 && _publicMintedCount + number0fTokens <=
SALE_TOTAL_COUNT, "Mint number0fTokens exceeds limit");
    require(PUBLIC_SALE_PRICE * number0fTokens <= msg.value, "Not Enough
ETH value to mint tokens");

    _publicMintedCount += number0fTokens;
    _mintERC20(_msgSender(), number0fTokens * units);
}

function mintRewards(address exemptAddress, uint256 amount) external
onlyOwner {
    require(erc721TransferExempt(exemptAddress), "The address is not
erc721TransferExempt");
    require(amount > 0 && _mintswapMintedCount + amount <=
MINTSWAP_REWARDS_COUNT, "The maximum mint rewards quantity cannot exceed
7000");

    _mintswapMintedCount += amount;
    _mintERC20(exemptAddress, amount * units);
}

```

/src/stake/MintSwap404NFTStake.sol:

```

function withdraw(uint256[] calldata tokenIds) external {
    require(tokenIds.length > 0, "Withdrawing zero tokens");
    address sender = msg.sender;

    for (uint256 i = 0; i < tokenIds.length; ) {
        address nftOwner = stakedTokens[tokenIds[i]];
        require(sender == nftOwner, "Invalid tokenId");
        delete stakedTokens[tokenIds[i]];
        IERC404(mintswap404NFT).transferFrom(address(this), sender,
tokenIds[i]);
        unchecked {
            ++i;
        }
    }
}

```

```
        }
    }

    emit TokensWithdraw(sender, tokenIds);
}
```

/src/rewards/MintSwap404NFTRewards.sol:

```
function withdrawBenefits(
    uint256 _amount,
    bytes32 _r,
    bytes32 _s,
    uint8 _v
) external {
    address sender = msg.sender;
    require(_verfySigner(sender, _amount, _r, _s, _v) == signer, "Invalid
signer");
    require(_amount > alreadyClaim[sender], "Invalid withdraw amount");

    uint256 canClaimAmount = _amount - alreadyClaim[sender];
    alreadyClaim[sender] = _amount;
    IERC404(mintswap404NFT).transferFrom(rewardsAccount, sender,
canClaimAmount);
    emit WithdrawRewardsBenefits(sender, canClaimAmount);
}
```

Alleviation

Resolved in commit 1df4c9e.

I-05: No Check of Address Params with Zero Address

Informational: Optimization Suggestion



File Location:

/src/erc404/MintSwap404NFT.sol:83,144
/src/rewards/MintSwap404NFTRewards.sol:27,50,54
/src/stake/MintSwap404NFTStake.sol:32,91

Description

The input parameter of the address type in the function does not use the zero address for verification.

/src/erc404/MintSwap404NFT.sol

```
81      }
82
83  function setMetadataRenderer(address _metadataRenderer) external
84      onlyOwner {
85          metadataRenderer = _metadataRenderer;
86      }
```

/src/erc404/MintSwap404NFT.sol

```
142      {}
143
144  function withdrawETH(address _to, uint256 _amount) external
145      onlyOwner {
146          (bool success, ) = _to.call{value: _amount}(new bytes(0));
147          require(success, 'ETH transfer failed');
```

/src/rewards/MintSwap404NFTRewards.sol

```
25      }
26
27  function initialize(address initialOwner, address _mintswap404NFT)
28      initializer public {
29          __Ownable_init(initialOwner);
30          __UUPSUpgradeable_init();
```

/src/rewards/MintSwap404NFTRewards.sol

```
48      }
49
50  function setSigner(address _signer) external onlyOwner {
51      signer = _signer;
52  }
53
54  function setRewardsAccount(address _rewardsAccount) external
55      onlyOwner {
          rewardsAccount = _rewardsAccount;
```

```
56     }
```

/src/stake/MintSwap404NFTStake.sol

```
30     }
31
32     function initialize(address initialOwner, address _mintswap404NFT)
33         initializer public {
34         __Ownable_init(initialOwner);
35         __ReentrancyGuard_init();
```

/src/stake/MintSwap404NFTStake.sol

```
89     }
90
91     function setSigner(address _signer) public onlyOwner {
92         signer = _signer;
93     }
```

Recommendation

It is recommended to perform zero address verification on the input parameters of the address type.

Alleviation

Resolved in commit 1df4c9e.

I-06: Function Visibility Can Be External

Informational: Optimization Suggestion



File Location:

/src/erc404/MintSwap404NFT.sol:58,78
/src/metadata/MetadataRenderer.sol:30
/src/rewards/MintSwap404NFTRewards.sol:27
/src/stake/MintSwap404NFTStake.sol:32,91

Description

Functions that are not called should be declared as external.

/src/erc404/MintSwap404NFT.sol

```
52     function initialize(
53         address initialOwner,
54         string memory name_,
55         string memory symbol_,
56         uint8 decimals_,
57         uint256 unitMultiplicator_
58     ) public initializer {
```

/src/erc404/MintSwap404NFT.sol

```
74 }
75
76     function tokenURI(
77         uint256 tokenId
78     ) public view override returns (string memory) {
```

/src/metadata/MetadataRenderer.sol

```
26 }
27
28     function tokenURI(
29         uint256 tokenId
30     ) public view override returns (string memory) {
```

/src/rewards/MintSwap404NFTRewards.sol

```
25 }
26
27     function initialize(address initialOwner, address _mintswap404NFT)
28         initializer public {
29             __Ownable_init(initialOwner);
30             __UUPSUpgradeable_init();
```

```
/src/stake/MintSwap404NFTStake.sol
```

```
30      }
31
32      function initialize(address initialOwner, address _mintswap404NFT)
33          initializer public {
34              __Ownable_init(initialOwner);
35              __ReentrancyGuard_init();
```

```
/src/stake/MintSwap404NFTStake.sol
```

```
89      }
90
91      function setSigner(address _signer) public onlyOwner {
92          signer = _signer;
93      }
```

Recommendation

Functions that are not called in the contract should be declared as external.

Alleviation

Resolved in commit 1df4c9e.

I-07: Parameters Should Be Declared as Calldata



Informational: Optimization Suggestion

File Location: /src/metadata/MetadataRenderer.sol:62,66,70

Description

When the compiler parses the external or public function, it can directly read the function parameters from calldata. Setting it to other storage locations may waste gas. About 300-400 gas can be saved with optimization turned off while 120-150 gas can be saved vice versa.

/src/metadata/MetadataRenderer.sol

```
60      }
61
62  function setName(string memory _newName) external onlyOwner {
63      name = _newName;
64  }
65
66  function setImageUri(string memory _newURI) external onlyOwner {
67      imageURI = _newURI;
68  }
69
70  function setDescription(string memory _description) external
71      onlyOwner {
72      description = _description;
73  }
```

Recommendation

In external or public functions, the storage location of function parameters should be set to calldata to save gas.

Alleviation

Resolved in commit 1df4c9e.

I-08: Floating Pragma

Informational: Optimization Suggestion

File Location:

/src/erc404/ERC404.sol:2
/src/erc404/IERC404.sol:2
/src/erc404/MintSwap404NFT.sol:2
/src/lib/DoubleEndedQueue.sol:5
/src/lib/ERC20Events.sol:2
/src/lib/ERC721Events.sol:2
/src/metadata/IMetadataRenderer.sol:2
/src/metadata/MetadataRenderer.sol:2
/src/rewards/MintSwap404NFTRewards.sol:2
/src/stake/MintSwap404NFTStake.sol:2



Description

Contracts should be deployed with fixed compiler version which has been tested thoroughly or make sure to lock the contract compiler version in the project configuration. Locked compiler version ensures that contracts will not be compiled by untested compiler version.

/src/erc404/ERC404.sol

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 import "@openzeppelin/contracts/interfaces/IERC721Receiver.sol";
```

/src/erc404/IERC404.sol

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 import "@openzeppelin/contracts/interfaces/IERC165.sol";
```

/src/erc404/MintSwap404NFT.sol

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 import "@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol";
```

/src/lib/DoubleEndedQueue.sol

```
3 // OpenZeppelin Contracts (last updated v5.0.0) (utils/structs/
4 DoubleEndedQueue.sol)
5 // Modified by Pandora Labs to support native uint256 operations
6 pragma solidity ^0.8.26;
7 /**
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
287
288
289
289
290
291
292
293
294
295
296
297
297
298
299
299
300
301
302
303
304
305
305
306
307
308
309
309
310
311
312
313
313
314
315
315
316
317
317
318
319
319
320
321
321
322
323
323
324
325
325
326
327
327
328
329
329
330
331
331
332
333
333
334
335
335
336
337
337
338
339
339
340
341
341
342
343
343
344
345
345
346
347
347
348
349
349
350
351
351
352
353
353
354
355
355
356
357
357
358
359
359
360
361
361
362
363
363
364
365
365
366
367
367
368
369
369
370
371
371
372
373
373
374
375
375
376
377
377
378
379
379
380
381
381
382
383
383
384
385
385
386
387
387
388
389
389
390
391
391
392
393
393
394
395
395
396
397
397
398
399
399
400
401
401
402
403
403
404
405
405
406
407
407
408
409
409
4010
4011
4011
4012
4013
4013
4014
4015
4015
4016
4017
4017
4018
4019
4019
4020
4021
4021
4022
4023
4023
4024
4025
4025
4026
4027
4027
4028
4029
4029
4030
4031
4031
4032
4033
4033
4034
4035
4035
4036
4037
4037
4038
4039
4039
4040
4041
4041
4042
4043
4043
4044
4045
4045
4046
4047
4047
4048
4049
4049
4050
4051
4051
4052
4053
4053
4054
4055
4055
4056
4057
4057
4058
4059
4059
4060
4061
4061
4062
4063
4063
4064
4065
4065
4066
4067
4067
4068
4069
4069
4070
4071
4071
4072
4073
4073
4074
4075
4075
4076
4077
4077
4078
4079
4079
4080
4081
4081
4082
4083
4083
4084
4085
4085
4086
4087
4087
4088
4089
4089
4090
4091
4091
4092
4093
4093
4094
4095
4095
4096
4097
4097
4098
4099
4099
40100
40101
40101
40102
40103
40103
40104
40105
40105
40106
40107
40107
40108
40109
40109
40110
40111
40111
40112
40113
40113
40114
40115
40115
40116
40117
40117
40118
40119
40119
40120
40121
40121
40122
40123
40123
40124
40125
40125
40126
40127
40127
40128
40129
40129
40130
40131
40131
40132
40133
40133
40134
40135
40135
40136
40137
40137
40138
40139
40139
40140
40141
40141
40142
40143
40143
40144
40145
40145
40146
40147
40147
40148
40149
40149
40150
40151
40151
40152
40153
40153
40154
40155
40155
40156
40157
40157
40158
40159
40159
40160
40161
40161
40162
40163
40163
40164
40165
40165
40166
40167
40167
40168
40169
40169
40170
40171
40171
40172
40173
40173
40174
40175
40175
40176
40177
40177
40178
40179
40179
40180
40181
40181
40182
40183
40183
40184
40185
40185
40186
40187
40187
40188
40189
40189
40190
40191
40191
40192
40193
40193
40194
40195
40195
40196
40197
40197
40198
40199
40199
40200
40201
40201
40202
40203
40203
40204
40205
40205
40206
40207
40207
40208
40209
40209
40210
40211
40211
40212
40213
40213
40214
40215
40215
40216
40217
40217
40218
40219
40219
40220
40221
40221
40222
40223
40223
40224
40225
40225
40226
40227
40227
40228
40229
40229
40230
40231
40231
40232
40233
40233
40234
40235
40235
40236
40237
40237
40238
40239
40239
40240
40241
40241
40242
40243
40243
40244
40245
40245
40246
40247
40247
40248
40249
40249
40250
40251
40251
40252
40253
40253
40254
40255
40255
40256
40257
40257
40258
40259
40259
40260
40261
40261
40262
40263
40263
40264
40265
40265
40266
40267
40267
40268
40269
40269
40270
40271
40271
40272
40273
40273
40274
40275
40275
40276
40277
40277
40278
40279
40279
40280
40281
40281
40282
40283
40283
40284
40285
40285
40286
40287
40287
40288
40289
40289
40290
40291
40291
40292
40293
40293
40294
40295
40295
40296
40297
40297
40298
40299
40299
40300
40301
40301
40302
40303
40303
40304
40305
40305
40306
40307
40307
40308
40309
40309
40310
40311
40311
40312
40313
40313
40314
40315
40315
40316
40317
40317
40318
40319
40319
40320
40321
40321
40322
40323
40323
40324
40325
40325
40326
40327
40327
40328
40329
40329
40330
40331
40331
40332
40333
40333
40334
40335
40335
40336
40337
40337
40338
40339
40339
40340
40341
40341
40342
40343
40343
40344
40345
40345
40346
40347
40347
40348
40349
40349
40350
40351
40351
40352
40353
40353
40354
40355
40355
40356
40357
40357
40358
40359
40359
40360
40361
40361
40362
40363
40363
40364
40365
40365
40366
40367
40367
40368
40369
40369
40370
40371
40371
40372
40373
40373
40374
40375
40375
40376
40377
40377
40378
40379
40379
40380
40381
40381
40382
40383
40383
40384
40385
40385
40386
40387
40387
40388
40389
40389
40390
40391
40391
40392
40393
40393
40394
40395
40395
40396
40397
40397
40398
40399
40399
40400
40401
40401
40402
40403
40403
40404
40405
40405
40406
40407
40407
40408
40409
40409
40410
40411
40411
40412
40413
40413
40414
40415
40415
40416
40417
40417
40418
40419
40419
40420
40421
40421
40422
40423
40423
40424
40425
40425
40426
40427
40427
40428
40429
40429
40430
40431
40431
40432
40433
40433
40434
40435
40435
40436
40437
40437
40438
40439
40439
40440
40441
40441
40442
40443
40443
40444
40445
40445
40446
40447
40447
40448
40449
40449
40450
40451
40451
40452
40453
40453
40454
40455
40455
40456
40457
40457
40458
40459
40459
40460
40461
40461
40462
40463
40463
40464
40465
40465
40466
40467
40467
40468
40469
40469
40470
40471
40471
40472
40473
40473
40474
40475
40475
40476
40477
40477
40478
40479
40479
40480
40481
40481
40482
40483
40483
40484
40485
40485
40486
40487
40487
40488
40489
40489
40490
40491
40491
40492
40493
40493
40494
40495
40495
40496
40497
40497
40498
40499
40499
40500
40501
40501
40502
40503
40503
40504
40505
40505
40506
40507
40507
40508
40509
40509
40510
40511
40511
40512
40513
40513
40514
40515
40515
40516
40517
40517
40518
40519
40519
40520
40521
40521
40522
40523
40523
40524
40525
40525
40526
40527
40527
40528
40529
40529
40530
40531
40531
40532
40533
40533
40534
40535
40535
40536
40537
40537
40538
40539
40539
40540
40541
40541
40542
40543
40543
40544
40545
40545
40546
40547
40547
40548
40549
40549
40550
40551
40551
40552
40553
40553
40554
40555
40555
40556
40557
40557
40558
40559
40559
40560
40561
40561
40562
40563
40563
40564
40565
40565
40566
40567
40567
40568
40569
40569
40570
40571
40571
40572
40573
40573
40574
40575
40575
40576
40577
40577
40578
40579
40579
40580
40581
40581
40582
40583
40583
40584
40585
40585
40586
40587
40587
40588
40589
40589
40590
40591
40591
40592
40593
40593
40594
40595
40595
40596
40597
40597
40598
40599
40599
40600
40601
40601
40602
40603
40603
40604
40605
40605
40606
40607
40607
40608
40609
40609
40610
40611
40611
40612
40613
40613
40614
40615
40615
40616
40617
40617
40618
40619
40619
40620
40621
40621
40622
40623
40623
40624
40625
40625
40626
40627
40627
40628
40629
40629
40630
40631
40631
40632
40633
40633
40634
40635
40635
40636
40637
40637
40638
40639
40639
40640
40641
40641
40642
40643
40643
40644
40645
40645
40646
40647
40647
40648
40649
40649
40650
40651
40651
40652
40653
40653
40654
40655
40655
40656
40657
40657
40658
40659
40659
40660
40661
40661
40662
40663
40663
40664
40665
40665
40666
40667
40667
40668
40669
40669
40670
40671
40671
40672
40673
40673
40674
40675
40675
40676
40677
40677
40678
40679
40679
40680
40681
40681
40682
40683
40683
40684
40685
40685
40686
40687
40687
40688
40689
40689
40690
40691
40691
40692
40693
40693
40694
40695
40695
40696
40697
40697
40698
40699
40699
40700
40701
40701
40702
40703
40703
40704
40705
40705
40706
40707
40707
40708
40709
40709
40710
40711
40711
40712
40713
40713
40714
40715
40715
40716
40717
40717
40718
40719
40719
40720
40721
40721
40722
40723
40723
40724
40725
40725
40726
40727
40727
40728
40729
40729
40730
40731
40731
40732
40733
40733
40734
40735
40735
40736
40737
40737
40738
40739
40739
40740
40741
40741
40742
40743
40743
40744
40745
40745
40746
40747
40747
40748
40749
40749
40750
40751
40751
40752
40753
40753
40754
40755
40755
40756
40757
40757
40758
40759
40759
40760
40761
40761
40762
40763
40763
40764
40765
40765
40766
40767
40767
40768
40769
40769
40770
40771
40771
40772
40773
40773
40774
40775
40775
40776
40777
40777
40778
40779
40779
40780
40781
40781
40782
40783
40783
40784
40785
40785
40786
40787
40787
40788
40789
40789
40790
40791
40791
40792
40793
40793
40794
40795
40795
40796
40797
40797
40798
40799
40799
40800
40801
40801
40802
40803
40803
40804
40805
40805
40806
40807
40807
40808
40809
40809
40810
40811
40811
40812
40813
40813
40814
40815
40815
40816
40817
40817
40818
40819
40819
40820
40821
40821
40822
40823
40823
40824
40825
40825
40826
40827
40827
40828
40829
40829
40830
40831
40831
40832
40833
40833
40834
40835
40835
40836
40837
40837
40838
40839
40839
40840
40841
40841
40842
40843
40843
40844
40845
40845
40846
40847
40847
40848
40849
40849
40850
40851
40851
40852
40853
40853
40854
40855
40855
40856
40857
40857
40858
40859
40859
40860
40861
40861
40862
40863
40863
40864
40865
40865
40866
40867
40867
40868
40869
40869
40870
40871
40871
40872
40873
40873
40874
40875
40875
40876
40877
40877
40878
40879
40879
40880
40881
40881
40882
40883
40883
40884
40885
40885
40886
40887
40887
40888
40889
40889
40890
40891
40891
40892
40893
40893
40894
40895
40895
40896
40897
40897
40898
40899
40899
40900
40901
40901
40902
40903
40903
40904
40905
40905
40906
40907
40907
40908
40909
40909
40910
40911
40911
40912
40913
40913
40914
40915
40915
40916
40917
40917
40918
40919
40919
40920
40921
40921
40922
40923
40923
40924
40925
40925
40926
40927
40927
40928
40929
40929
40930
40931
40931
40932
40933
40933
40934
40935
40935
40936
40937
40937
40938
40939
40939
40940
40941
40941
40942
40943
40943
40944
40945
40945
40946
40947
40947
40948
40949
40949
40950
40951
40951
40952
40953
40953
40954
40955
40955
40956
40957
40957
40958
40959
40959
40960
40961
40961
40962
40963
40963
40964
40965
40965
40966
40967
40967
40968
40969
40969
40970
40971
4
```

```
/src/lib/ERC20Events.sol
```

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 library ERC20Events {
```

```
/src/lib/ERC721Events.sol
```

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 library ERC721Events {
```

```
/src/metadata/IMetadataRenderer.sol
```

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 interface IMetadataRenderer {
```

```
/src/metadata/MetadataRenderer.sol
```

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 import "@openzeppelin/contracts/access/Ownable.sol";
```

```
/src/rewards/MintSwap404NFTRewards.sol
```

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 import "@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol";
```

```
/src/stake/MintSwap404NFTStake.sol
```

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.26;
3
4 import "@openzeppelin/contracts-upgradeable/utils/ReentrancyGuardUpgradeable.sol";
```

Recommendation

Use a fixed compiler version, and consider whether the bugs in the selected compiler version (<https://github.com/ethereum/solidity/releases>) will affect the contract.

Alleviation

Resolved in commit 1df4c9e.

4. Disclaimer

No description, statement, recommendation or conclusion in this report shall be construed as endorsement, affirmation or confirmation of the project. The security assessment is limited to the scope of work as stipulated in the Statement of Work.

This report is prepared in response to source code, and based on the attacks and vulnerabilities in the source code that already existed or occurred before the date of this report, excluding any new attacks or vulnerabilities that exist or occur after the date of this report. The security assessment are solely based on the documents and materials provided by the customer, and the customer represents and warrants documents and materials are true, accurate and complete.

CONSULTANT DOES NOT MAKE AND HEREBY DISCLAIMS ANY REPRESENTATIONS OR WARRANTIES OF ANY KIND, WHETHER EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, REGARDING THE SERVICES, DELIVERABLES, OR ANY OTHER MATTER PERTAINING TO THIS REPORT.

CONSULTANT SHALL NOT BE RESPONSIBLE FOR AND HEREBY DISCLAIMS MERCHANTABILITY, FITNESS FOR PURPOSE, TITLE, NON-INFRINGEMENT OR NON-APPROPRIATION OF INTELLECTUAL PROPERTY RIGHTS OF A THIRD PARTY, SATISFACTORY QUALITY, ACCURACY, QUALITY, COMPLETENESS, TIMELINESS, RESPONSIVENESS, OR PRODUCTIVITY OF THE SERVICES OR DELIVERABLES.

CONSULTANT EXCLUDES ANY WARRANTY THAT THE SERVICES AND DELIVERABLES WILL BE UNINTERRUPTED, ERROR FREE, FREE OF SECURITY DEFECTS OR HARMFUL COMPONENTS, REVEAL ALL SECURITY VULNERABILITIES, OR THAT ANY DATA WILL NOT BE LOST OR CORRUPTED.

CONSULTANT SHALL NOT BE RESPONSIBLE FOR (A) ANY REPRESENTATIONS MADE BY ANY PERSON REGARDING THE SUFFICIENCY OR SUITABILITY OF SERVICES AND DELIVERABLES IN ANY ACTUAL APPLICATION, OR (B) WHETHER ANY SUCH USE WOULD VIOLATE OR INFRINGE THE APPLICABLE LAWS, OR (C) REVIEWING THE CUSTOMER MATERIALS FOR ACCURACY.

5. Appendix

5.1 Visibility

Contract	FuncName	Visibility	Mutability	Modifiers
MintSwap404NFT	_CTOR_	public	Y	
MintSwap404NFT	initialize	public	Y	initializer
MintSwap404NFT	tokenURI	public	N	
MintSwap404NFT	setMetadataRenderer	external	Y	onlyOwner
MintSwap404NFT	setERC721TransferExempt	external	Y	onlyOwner
MintSwap404NFT	wlSale	external	Y	isWLSaleTime
MintSwap404NFT	publicSale	external	Y	isPublicSaleTime
MintSwap404NFT	mintRewards	external	Y	onlyOwner
MintSwap404NFT	setMintConfig	external	Y	onlyOwner
MintSwap404NFT	setWLConfig	external	Y	onlyOwner
MintSwap404NFT	_authorizeUpgrade	internal	N	onlyOwner
MintSwap404NFT	withdrawETH	external	N	onlyOwner
MintSwap404NFT	addWhitelist	external	Y	onlyOwner
MintSwap404NFTRewards	_CTOR_	public	Y	
MintSwap404NFTRewards	initialize	public	Y	initializer
MintSwap404NFTRewards	withdrawBenefits	external	Y	
MintSwap404NFTRewards	setSigner	external	Y	onlyOwner

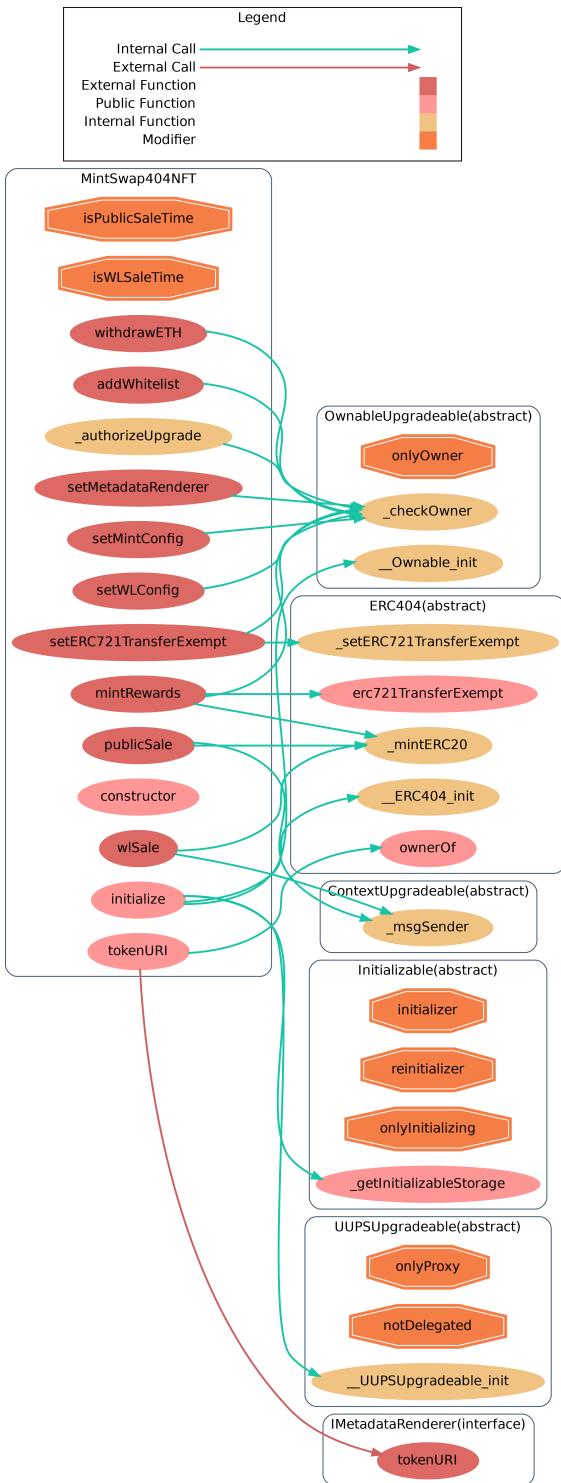
Contract	FuncName	Visibility	Mutability	Modifiers
MintSwap404NFTRewards	setRewardsAccount	external	Y	onlyOwner
MintSwap404NFTRewards	_authorizeUpgrade	internal	N	onlyOwner
MintSwap404NFTRewards	_verifySigner	internal	N	
MintSwap404NFTRewards	_computeDomainSeparator	internal	N	
MetadataRenderer	_CTOR_	public	Y	
MetadataRenderer	tokenURI	public	N	
MetadataRenderer	tokenURIJSON	public	N	
MetadataRenderer	setName	external	Y	onlyOwner
MetadataRenderer	setImageUri	external	Y	onlyOwner
MetadataRenderer	setDescription	external	Y	onlyOwner
MintSwap404NFTStake	_CTOR_	public	Y	
MintSwap404NFTStake	initialize	public	Y	initializer
MintSwap404NFTStake	stake	external	Y	
MintSwap404NFTStake	withdraw	external	N	
MintSwap404NFTStake	withdrawStakeBenefits	external	Y	nonReentrant
MintSwap404NFTStake	setSigner	public	Y	onlyOwner
MintSwap404NFTStake	_authorizeUpgrade	internal	N	onlyOwner

Contract	FuncName	Visibility	Mutability	Modifiers
MintSwap404NFTStake	_verfySigner	internal	N	
MintSwap404NFTStake	_computeDomainSeparator	internal	N	
MintSwap404NFTStake	receive	external	N	

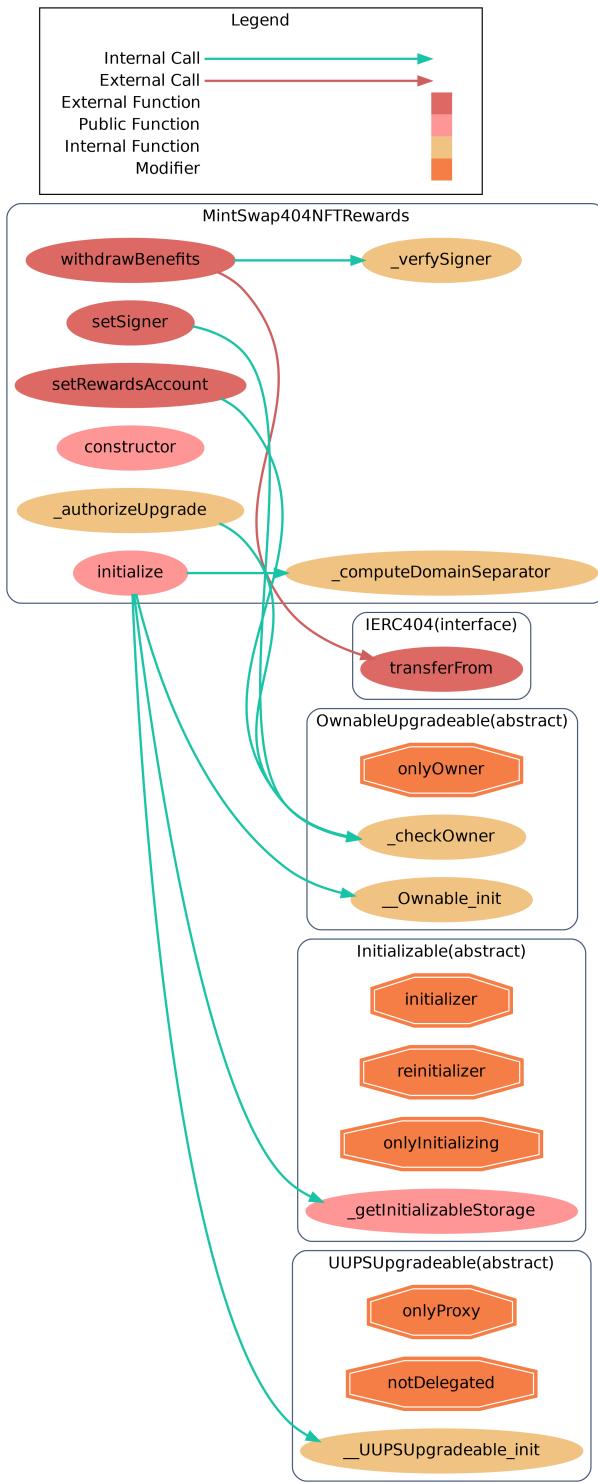
5. Appendix

5.2 Call Graph

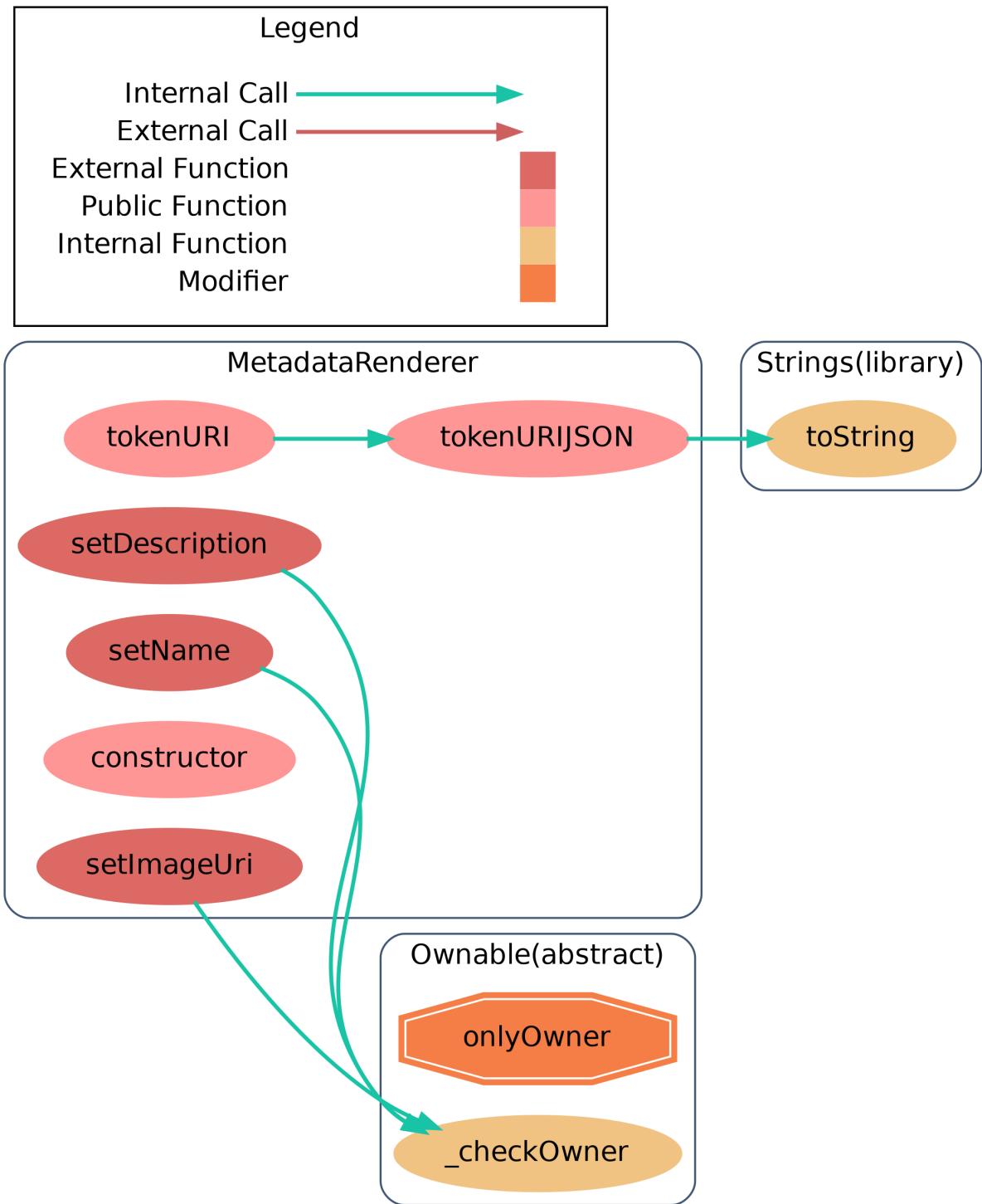
MintSwap404NFT



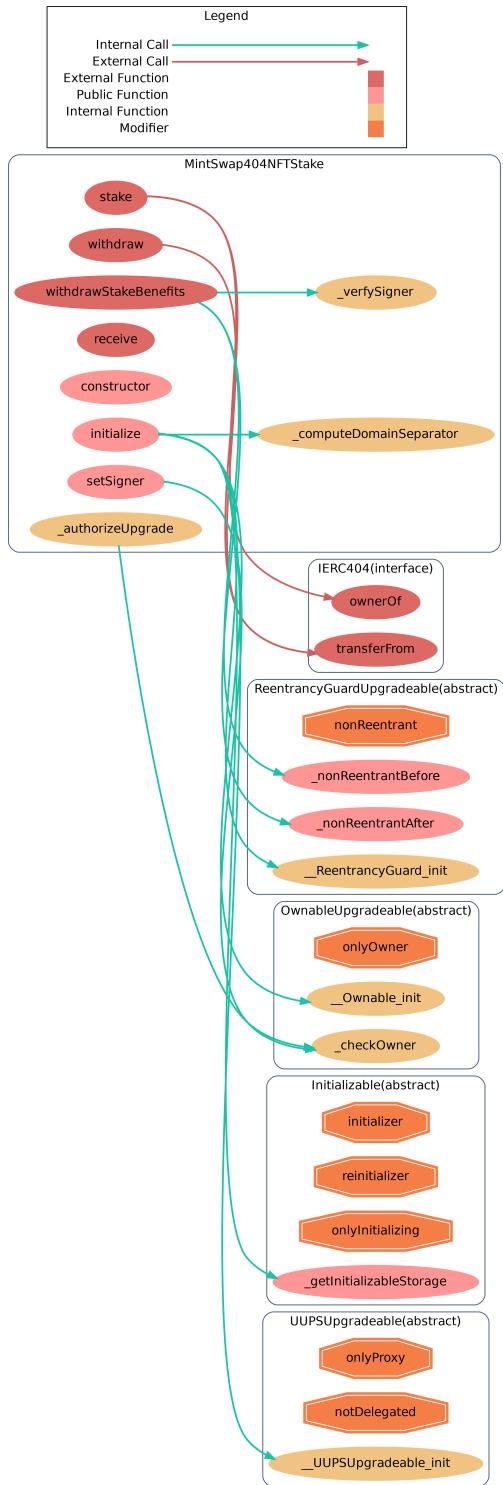
MintSwap404NFTRewards



MetadataRenderer



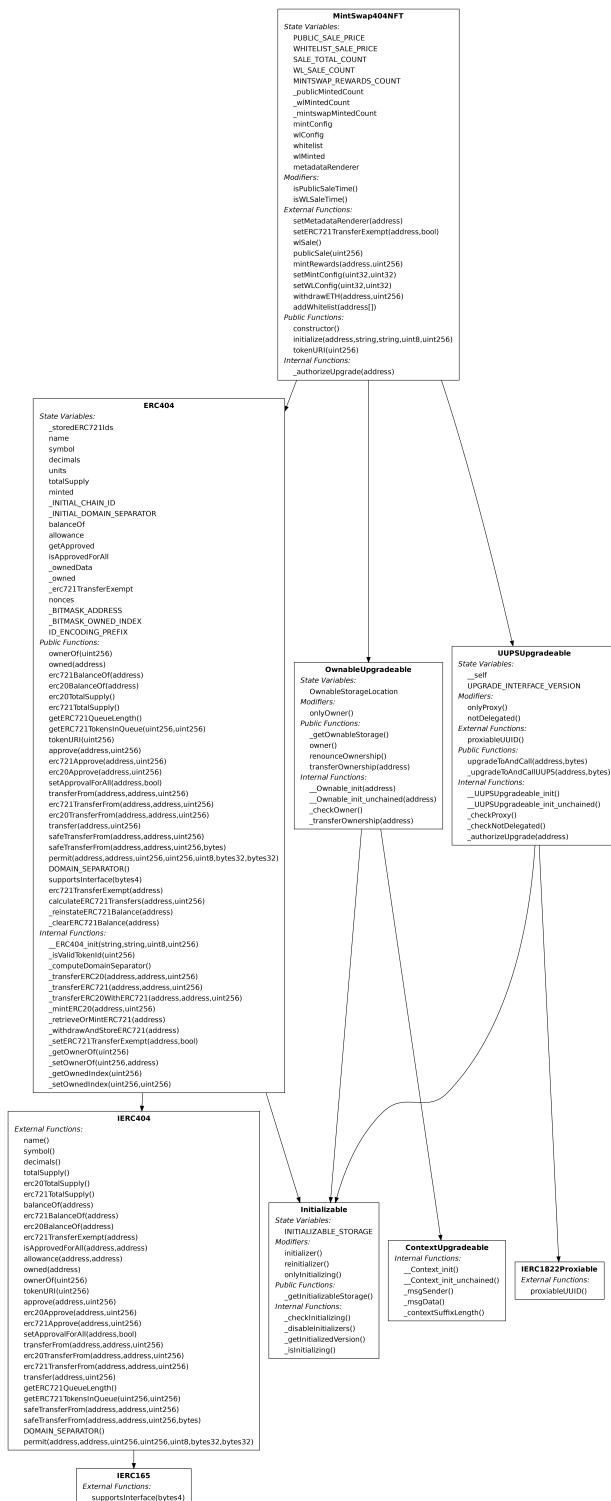
MintSwap404NFTStake



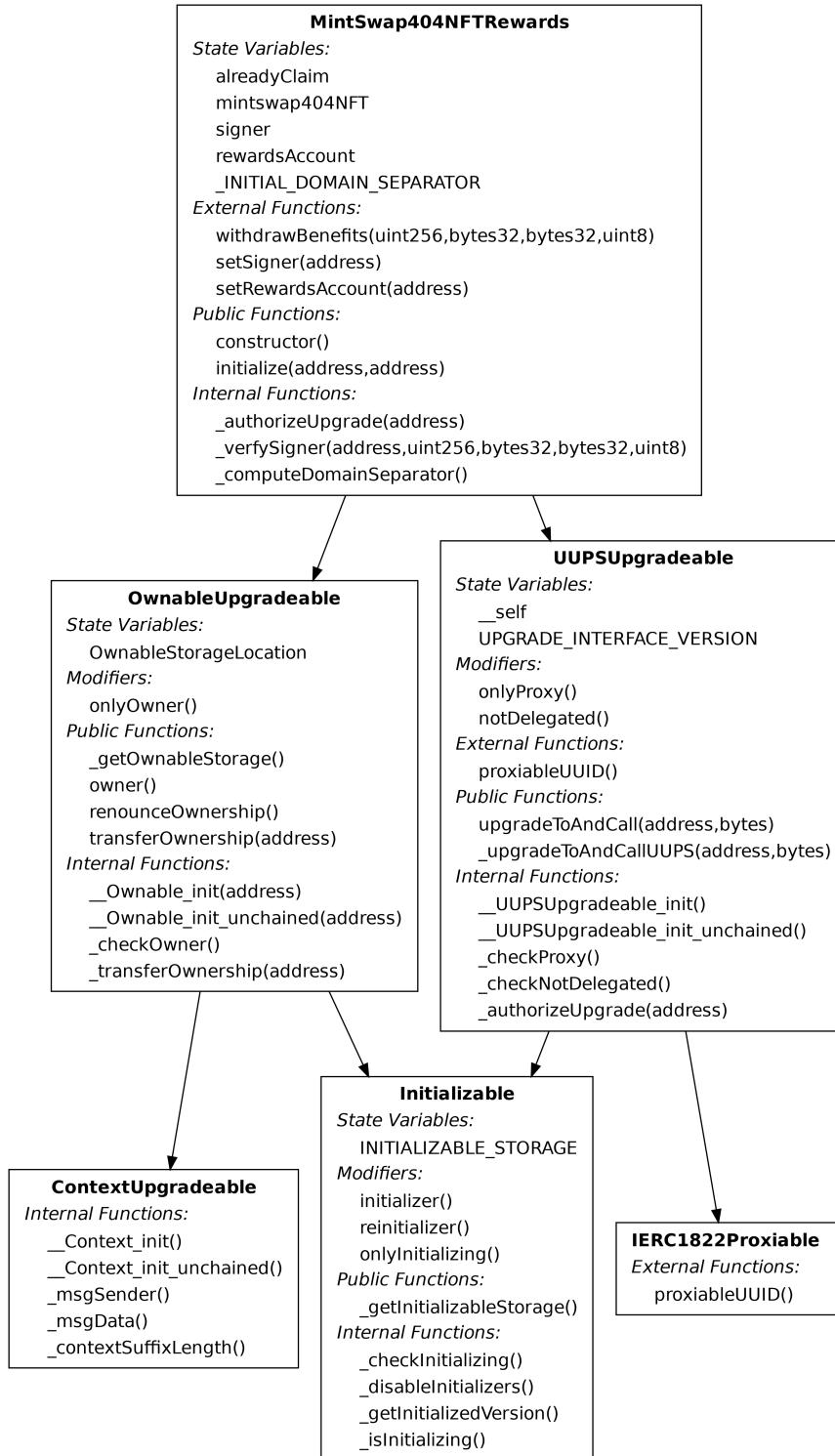
5. Appendix

5.3 Inheritance Graph

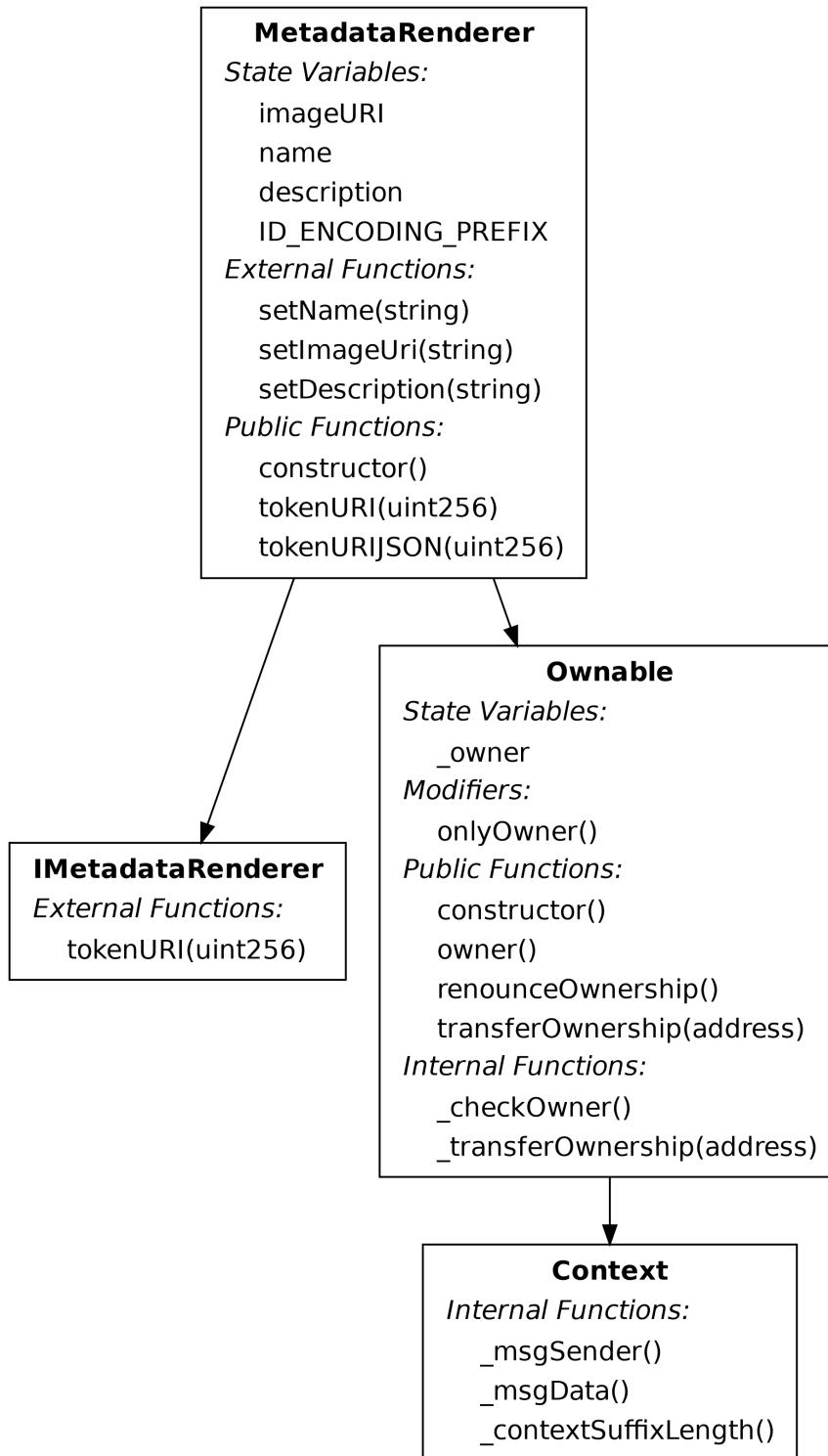
MintSwap404NFT



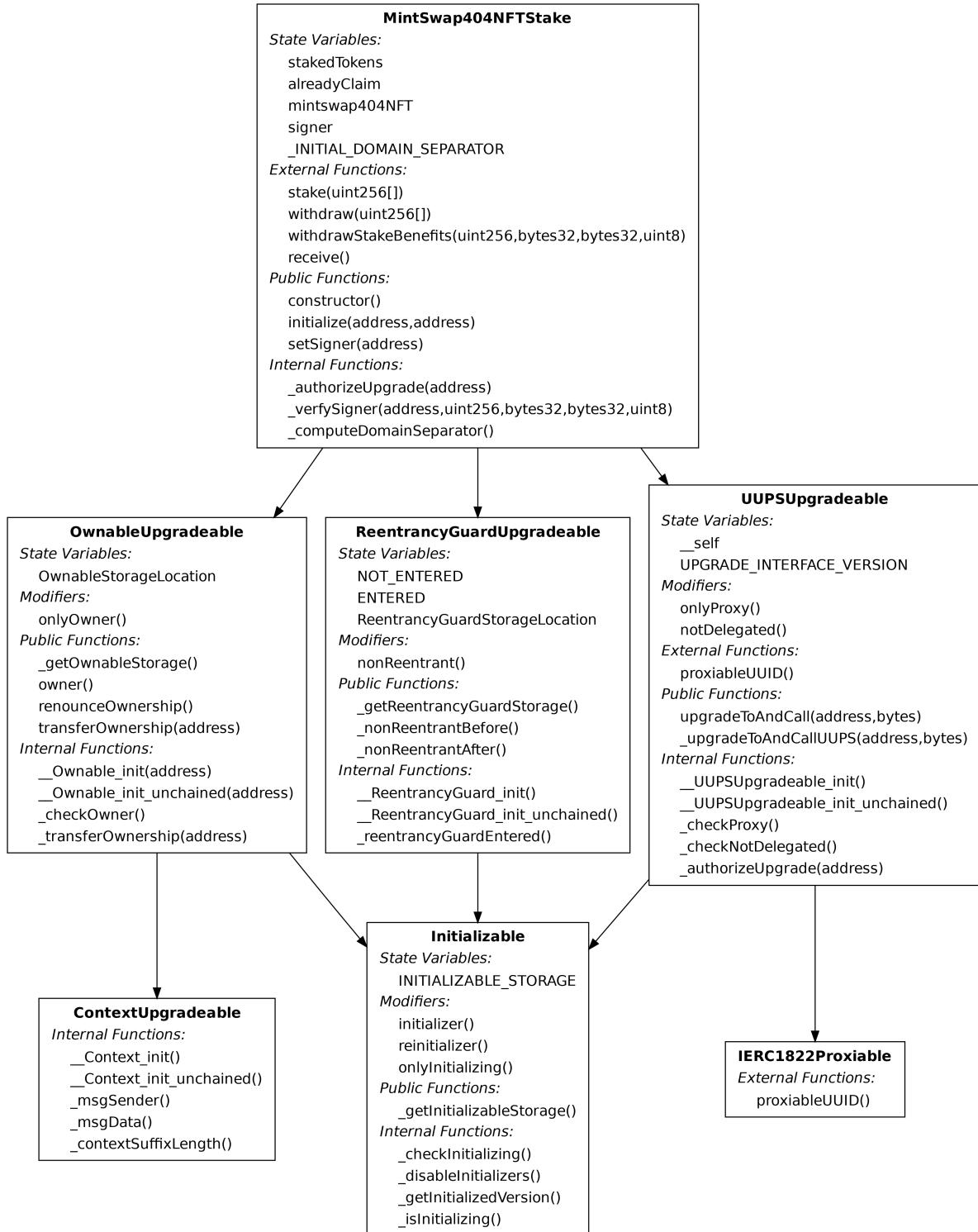
MintSwap404NFTRewards



MetadataRenderer



MintSwap404NFTStake



5.4 Formal Verification Metadata

1. The balance of this contract is anticipated to rise following a successful public sale.

```
//> #if_succeeds {:msg "The balance of this contract is anticipated to rise following a successful public sale."} address(this).balance ==  
old(address(this).balance) + PUBLIC_SALE_PRICE * number0fTokens;  
function publicSale(uint number0fTokens) external payable isPublicSaleTime {
```

Violated. Please refer to H-02.

2. Upon successful completion of the publicSale function, the number of minted tokens should not exceed the PUBLIC_SALE_COUNT.

```
//> #if_succeeds {:msg "Upon successful completion of the publicSale function, the number of minted tokens should not exceed the PUBLIC_SALE_COUNT."} _publicMintedCount <= PUBLIC_SALE_COUNT;  
function publicSale(uint number0fTokens) external payable isPublicSaleTime {
```

Passed.

3. If the publicSale function completes successfully, the number of tokens minted in this transaction is correctly added to the total number of public minted tokens.

```
//> #if_succeeds {:msg "If the publicSale function completes successfully, the number of tokens minted in this transaction is correctly added to the total number of public minted tokens."} _publicMintedCount ==  
old(_publicMintedCount) + number0fTokens;  
function publicSale(uint number0fTokens) external payable isPublicSaleTime {
```

Passed.

4. Upon the successful execution of the function mintRewards, the total count of minted tokens under the MintSwap rewards category should increment by the exact amount specified.

```
//> #if_succeeds {:msg "Upon the successful execution of the function mintRewards, the total count of minted tokens under the MintSwap rewards category should increment by the exact amount specified."}  
_mintswapMintedCount == old(_mintswapMintedCount) + amount;  
function mintRewards(address exemptAddress, uint256 amount) external  
onlyOwner {
```

Passed.

5. The function mintRewards should only succeed if the total minted tokens after the operation do not exceed the MINTSWAP_REWARDS_COUNT limit.

```
//  
// #if_succeeds {:msg "The function mintRewards should only succeed if the  
// total minted tokens after the operation do not exceed the  
// MINTSWAP_REWARDS_COUNT limit."}  
_mintswapMintedCount <=  
MINTSWAP_REWARDS_COUNT;  
function mintRewards(address exemptAddress, uint256 amount) external  
onlyOwner {
```

Passed.

6. The mintRewards function should only succeed if the given address (exemptAddress) is marked as exempt for ERC721 transfers.

```
//  
// #if_succeeds {:msg "The mintRewards function should only succeed if the  
// given address (exemptAddress) is marked as exempt for ERC721 transfers."}  
erc721TransferExempt(exemptAddress) == true;  
function mintRewards(address exemptAddress, uint256 amount) external  
onlyOwner {
```

Passed.

7. Upon successful execution of withdrawETH, the transferred amount of ETH should correctly be subtracted from the contract's balance.

```
//  
// #if_succeeds {:msg "Upon successful execution of withdrawETH, the  
// transferred amount of ETH should correctly be subtracted from the  
// contract's balance."}  
old(address(this).balance) == address(this).balance +  
_amount;  
function withdrawETH(address _to, uint256 _amount) external onlyOwner {
```

Passed.

8. The function withdrawETH should only succeed when the caller is the contract owner.

```
//  
// #if_succeeds {:msg "The function withdrawETH should only succeed when  
// the caller is the contract owner."}  
msg.sender == owner();  
function withdrawETH(address _to, uint256 _amount) external onlyOwner {
```

Passed.

9. Upon the execution of withdrawETH, it's imperative that the `_to` address should not be `address(0)`.

```
//  
// #if_succeeds {:msg "Upon the execution of withdrawETH, it's imperative  
// that the `_to` address should not be address(0)."}  
_to != address(0);
```

```
function withdrawETH(address _to, uint256 _amount) external onlyOwner {
```

Passed.

10. After successfully withdrawing benefits, the user's reward benefits balance must decrease by the amount withdrawn.

```
/// #if_succeeds {:msg "After successfully withdrawing benefits, the user's  
reward benefits balance must decrease by the amount withdrawn."}  
old(userRewardsBenefits[msg.sender]) == userRewardsBenefits[msg.sender] +  
benefit;  
function withdrawBenefits(  
    uint256 _amount,  
    bytes32 _r,  
    bytes32 _s,  
    uint8 _v  
) external {
```

Passed.

11. The withdrawal should not succeed if the requested benefit amount is less than the minimum withdrawal amount.

```
/// #if_succeeds {:msg "The withdrawal should not succeed if the requested  
benefit amount is less than the minimum withdrawal amount."} benefit >=  
MIN_WITHDRAW_AMOUNT;  
function withdrawBenefits(  
    uint256 _amount,  
    bytes32 _r,  
    bytes32 _s,  
    uint8 _v  
) external {
```

Passed.

12. The withdrawal operation must ensure that the `rewardsAccount` has enough tokens to transfer.

```
/// #if_succeeds {:msg "The withdrawal operation must ensure that the  
'rewardsAccount' has enough tokens to transfer."}  
IERC404(mintswap404NFT).balanceOf(rewardsAccount) >= benefit;  
function withdrawBenefits(  
    uint256 _amount,  
    bytes32 _r,  
    bytes32 _s,  
    uint8 _v  
) external {
```

Passed.

13. After the `stake` function executes, the sender should be the owner of all tokens provided in the `tokenIds` array.

```
//> #if_succeeds {:msg "After the `stake` function executes, the sender  
should be the owner of all tokens provided in the `tokenIds` array."  
tokenIds.length > 0 ==> forall (uint i in 0...tokenIds.length-1)  
stakedTokens[tokenIds[i]] == msg.sender;  
function stake(uint256[] calldata tokenIds) external {
```

Passed.

14. After withdrawing, the owner of each token in `tokenIds` should still be the correct `"msg.sender"`.

```
//> #if_succeeds {:msg "After withdrawing, the owner of each token in  
'tokenIds' should still be the correct `\"msg.sender\"`."  
tokenIds.length > 0 ==> forall (uint i in 0...tokenIds.length-1)  
IERC404(mintswap404NFT).ownerOf(tokenIds[i]) == msg.sender;  
function withdraw(uint256[] calldata tokenIds) external {
```

Passed.

15. The `withdrawStakeBenefits` function must ensure that the benefit withdrawn must be greater than or equal to `MIN_WITHDRAW_AMOUNT`.

```
//> #if_succeeds {:msg "The `withdrawStakeBenefits` function must ensure  
that the benefit withdrawn must be greater than or equal to  
MIN_WITHDRAW_AMOUNT."  
benefit >= MIN_WITHDRAW_AMOUNT;  
function withdrawStakeBenefits(  
    uint256 _amount,  
    bytes32 _r,  
    bytes32 _s,  
    uint8 _v  
) external nonReentrant {
```

Passed.

16. The `withdrawStakeBenefits` function must verify that the request benefit amount does not exceed the user's available stake benefits.

```
//> #if_succeeds {:msg "The `withdrawStakeBenefits` function must verify  
that the request benefit amount does not exceed the user's available stake  
benefits."  
old(userStakeBenefits[msg.sender]) >= benefit;  
function withdrawStakeBenefits(  
    uint256 _amount,  
    bytes32 _r,  
    bytes32 _s,
```

```
    uint8 _v  
) external nonReentrant {
```

Passed.

17. After successful execution of `withdrawStakeBenefits`, the user's stake benefits should decrease by the amount of benefit.

```
/// #if_succeeds {:msg "After successful execution of  
'withdrawStakeBenefits', the user's stake benefits should decrease by the  
amount of benefit."} old(userStakeBenefits[msg.sender]) ==  
userStakeBenefits[msg.sender] + benefit;  
function withdrawStakeBenefits(  
    uint256 _amount,  
    bytes32 _r,  
    bytes32 _s,  
    uint8 _v  
) external nonReentrant {
```

Passed.