



Rario

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **October 11th, 2021 – November 1st, 2021**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY	5
CONTACTS	5
1 EXECUTIVE OVERVIEW	6
1.1 INTRODUCTION	7
1.2 AUDIT SUMMARY	7
1.3 TEST APPROACH & METHODOLOGY	7
RISK METHODOLOGY	8
1.4 SCOPE	10
2 ASSESSMENT SUMMARY & FINDINGS OVERVIEW	11
3 FINDINGS & TECH DETAILS	12
3.1 (HAL-01) INCORRECT LOGIC LEADS TO DOS IN AUCTION SALES - HIGH	
14	
Description	14
Risk Level	16
Recommendation	16
Remediation Plan	17
3.2 (HAL-02) OUTBID USER DOES NOT RECEIVE BACK HIS FUNDS - HIGH	18
Description	18
Risk Level	18
Recommendation	19
Remediation Plan	19
3.3 (HAL-03) UNCHECKED TRANSFER - MEDIUM	20
Description	20
Code Location	20

Risk Level	20
Recommendation	20
Remediation Plan	20
3.4 (HAL-04) USE OF TRANSFER WHEN TRANSFERRING NFTS - LOW	21
Description	21
Risk Level	21
Recommendation	21
Remediation Plan	21
3.5 (HAL-05) UPDATEUSER FUNCTION MISSING REQUIRE STATEMENT - LOW	23
Description	23
Risk Level	24
Recommendation	24
Remediation Plan	24
3.6 (HAL-06) MINT FUNCTION IN RARIOPOLYGONNFT DOES NOT VERIFY IF TOKEN WAS WITHDRAWN - INFORMATIONAL	26
Description	26
Risk Level	27
Recommendation	27
Remediation Plan	27
3.7 (HAL-07) POSSIBLE MISUSE OF PUBLIC FUNCTIONS - INFORMATIONAL	28
Description	28
Risk Level	28
Recommendation	28

Remediation Plan	28
3.8 (HAL-08) UNUSED CONSTANTS - INFORMATIONAL	29
Description	29
Code Location	29
Risk Level	29
Recommendation	29
Remediation Plan	29
3.9 (HAL-09) UNUSED EVENTS - INFORMATIONAL	30
Description	30
Risk Level	30
Recommendation	31
Remediation Plan	31
4 MANUAL TESTING	32
4.1 INTRODUCTION	33
4.2 UPGRADEABLE CONTRACTS' ARCHITECTURE	34
4.3 CONTRACTS INITIALIZATION	36
4.4 RARIOCURRENCYMANAGER CONTRACT	38
4.5 RARIODATA CONTRACT	40
4.6 RARIOMARKETPLACE CONTRACT	42
4.7 RARIOMARKETPLACECONFIG CONTRACT	49
4.8 RARIOPOLYGONTOKEN CONTRACT	51
4.9 RARIOETHEREUMTOKEN CONTRACT	53
4.10 RARIOUSERS CONTRACT	55
5 AUTOMATED TESTING	57
5.1 STATIC ANALYSIS REPORT	58
Description	58

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	10/11/2021	Roberto Reigada
0.2	Document Edits	11/01/2021	Roberto Reigada
0.3	Document Review	11/02/2021	Gabi Urrutia
1.0	Remediation Plan	11/08/2021	Roberto Reigada
1.1	Remediation Plan Review	11/09/2021	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Roberto Reigada	Halborn	Roberto.Reigada@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Rario engaged Halborn to conduct a security audit on their smart contracts beginning on October 11th, 2021 and ending on November 1st, 2021. The security assessment was scoped to the smart contracts provided in the Github repository [rariohq/mjolnir - v1 branch](#)

1.2 AUDIT SUMMARY

The team at Halborn was provided two weeks for the engagement and assigned a full time security engineer to audit the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues with the smart contracts

In summary, Halborn identified some security risks that were addressed by the [Rario team](#).

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the bridge code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into architecture and purpose
- Smart contract manual code review and walkthrough
- Graphing out functionality and contract logic/connectivity/functions ([solgraph](#))
- Manual assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes
- Manual testing by custom scripts
- Static Analysis of security for scoped contract, and imported functions. ([Slither](#))
- Testnet deployment ([Brownie](#), [Remix IDE](#))

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident, and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. It's quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that was used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of **5** to **1** with **5** being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

- 5 - Almost certain an incident will occur.
- 4 - High probability of an incident occurring.
- 3 - Potential of a security incident in the long term.
- 2 - Low probability of an incident occurring.
- 1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

- 5 - May cause devastating and unrecoverable impact or loss.
- 4 - May cause a significant level of impact or loss.
- 3 - May cause a partial impact or loss to many.
- 2 - May cause temporary impact or loss.

EXECUTIVE OVERVIEW

1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
----------	------	--------	-----	---------------

10 - CRITICAL

9 - 8 - HIGH

7 - 6 - MEDIUM

5 - 4 - LOW

3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

IN-SCOPE:

The security assessment was scoped to the following smart contracts:

- RarioCurrencyManager.sol
- RarioData.sol
- RarioEthereumToken.sol
- RarioMarketplace.sol
- RarioMarketplaceConfig.sol
- RarioPolygonToken.sol
- RarioUsers.sol
- traits/AccessControl.sol
- traits/Context.sol
- traits/CurrencyManager.sol
- traits/ERC165Interface.sol
- traits/ERC1822UUPS.sol
- traits/ERC1967Upgrade.sol
- traits/ERC721NFT.sol
- traits/Initializable.sol
- traits/Marketplace.sol
- traits/MarketplaceConfig.sol
- traits/MetadataDB.sol
- traits/Pausable.sol
- traits/RarioEthereumNFT.sol
- traits/RarioNFT.sol
- traits/RarioPolygonNFT.sol
- traits/ReentrancyGuard.sol
- traits/Upgradeable.sol
- traits/UserDB.sol
- All contracts inherited by these contracts

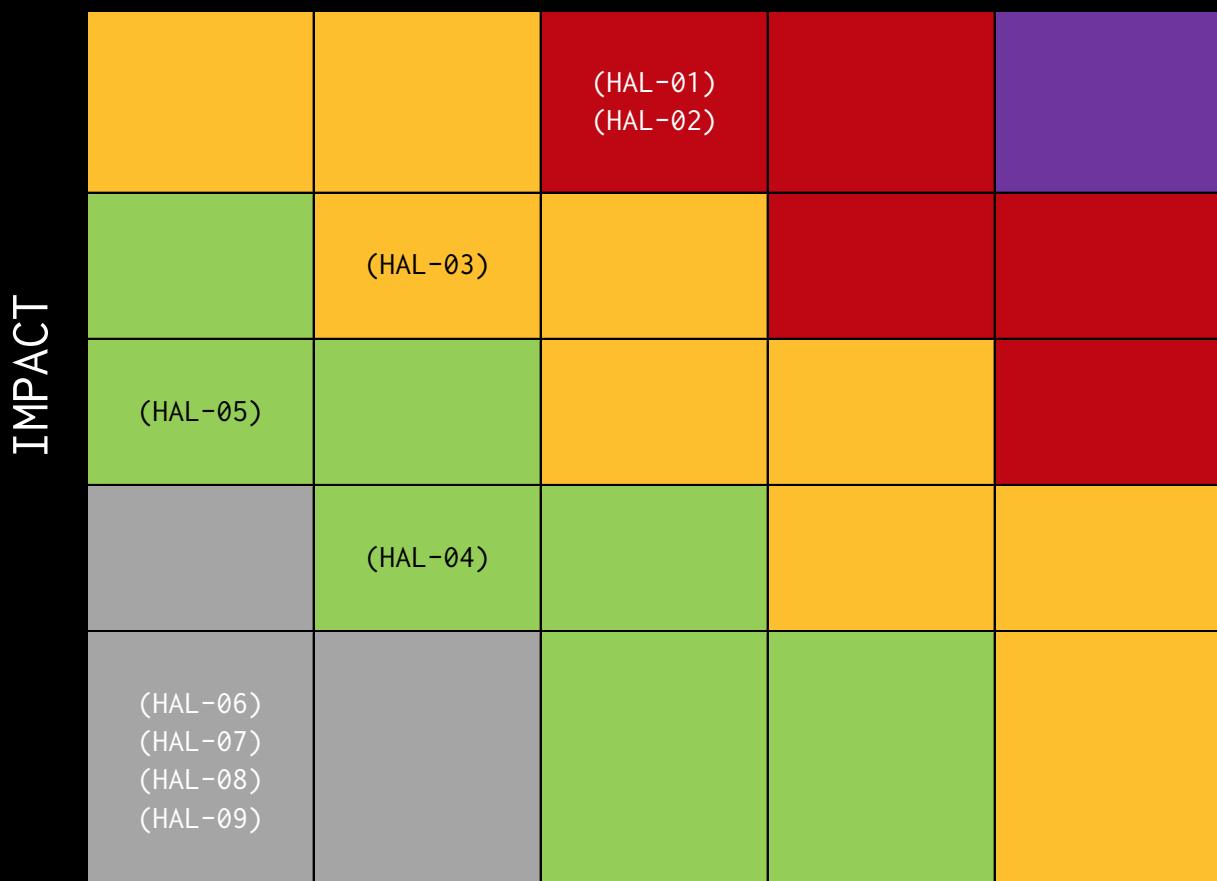
Commit ID: 2a16dcf44b02e3ecf90dcc6b05c8dff17e4f3aa

Fixed Commit ID: 01f0ba72f9067f6737a1d677e587959223f557e5

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	2	1	2	4

LIKELIHOOD

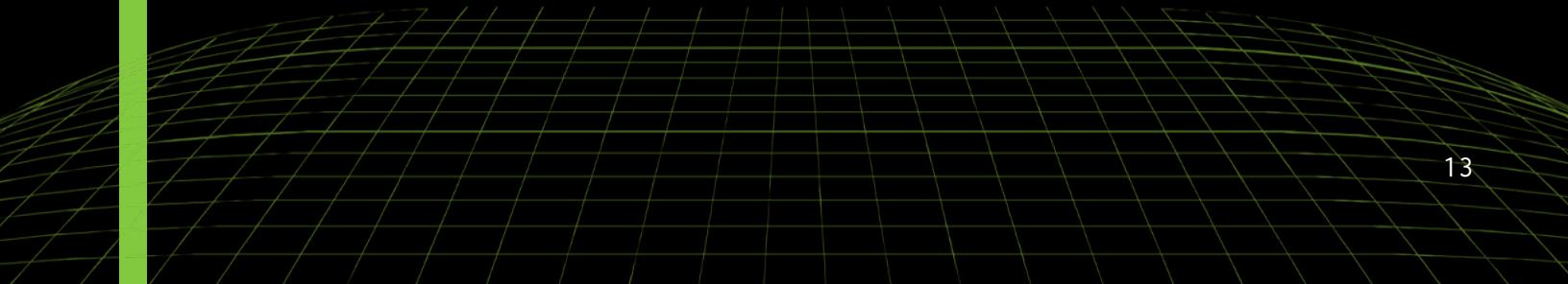


EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - INCORRECT LOGIC LEADS TO DOS IN AUCTION SALES	High	SOLVED - 11/08/2021
HAL02 - OUTBID USER DOES NOT RECEIVE BACK HIS FUNDS	High	SOLVED - 11/08/2021
HAL03 - UNCHECKED TRANSFER	Medium	SOLVED - 11/08/2021
HAL04 - USE OF TRANSFER WHEN TRANSFERRING NFTS	Low	SOLVED - 11/08/2021
HAL05 - UPDATEUSER FUNCTION MISSING REQUIRE STATEMENT	Low	SOLVED - 11/08/2021
HAL06 - MINT FUNCTION IN RARIOPOLYGONNFT DOES NOT VERIFY IF TOKEN WAS WITHDRAWN	Informational	SOLVED - 11/08/2021
HAL07 - POSSIBLE MISUSE OF PUBLIC FUNCTIONS	Informational	SOLVED - 11/08/2021
HAL08 - UNUSED CONSTANTS	Informational	SOLVED - 11/08/2021
HAL09 - UNUSED EVENTS	Informational	SOLVED - 11/08/2021



FINDINGS & TECH DETAILS



3.1 (HAL-01) INCORRECT LOGIC LEADS TO DOS IN AUCTION SALES - HIGH

Description:

In the contract `Marketplace.sol`, the function `placeBid` is used to buy an NFT token listing. If the `listingType` is a `RESERVE_PRICE_AUCTION`, the function follows this code:

```
Listing 1: Marketplace.sol - placeBid (Lines 331,332,370,374)

330 else if (listing.listingType == ListingType.RESERVE_PRICE_AUCTION)
{
331     if (listing.endTime == 0) {
332         // If this is the first bid, ensure it's >= the reserve
333         price
334         require(
335             listing.price <= bidValueInListingCurrency,
336             "NFTMarketReserveAuction: Bid must be at least the
337             reserve price"
338         );
339         listing.price = bidValueInListingCurrency;
340         listing.buyerPaymentWallet = payable(msgSender());
341         listing.buyerTokenWallet = payable(buyerTokenWallet);
342         listing.bidCurrency = currency;
343         listing.bidAmount = bidValue;
344         // On the first bid, the endTime is now + duration
345         listing.endTime = block.timestamp + listing.duration;
346         emit AuctionBidPlaced(listing.listingId, msgSender(),
347             bidValue, listing.endTime);
348     } else {
349         // If this bid outbids another, confirm that the bid is at
350         // least x% greater than the last
351         require(listing.endTime >= block.timestamp, "
352             NFTMarketReserveAuction: Auction is over");
353         require(
354             listing.buyerPaymentWallet != msgSender(),
355             "NFTMarketReserveAuction: You already have an
356             outstanding bid"
357         );
358         uint256 minAmount = (listing.price * (BASIS_POINTS +
359             _auctionPercentIncrementInBasisPoints)) /
```

```
353         BASIS_POINTS;
354         require(bidValueInListingCurrency >= minAmount, "
355             NFTMarketReserveAuction: Bid amount too low");
356         // Cache and update bidder state before a possible
357         // reentrancy (via the value transfer)
358         uint256 originalAmount = listing.bidAmount;
359         address payable originalBidder = listing.
360             buyerPaymentWallet;
361         listing.price = bidValueInListingCurrency;
362         listing.bidCurrency = currency;
363         listing.bidAmount = bidValue;
364         listing.buyerPaymentWallet = payable(msgSender());
365         listing.buyerTokenWallet = payable(buyerTokenWallet);
366
367         // When a bid outbids another, check to see if a time
368         // extension should apply.
369         if (listing.endTime - block.timestamp < listing.
370             extensionDuration) {
371             listing.endTime = block.timestamp + listing.
372                 extensionDuration;
373         }
374
375         if (currencyType == 2) {
376             IERC20(currencyContractAddress).transferFrom(msgSender
377                 (), address(this), bidValue);
378         }
379
380         emit AuctionBidPlaced(listing.listingId, msgSender(),
381             bidValue, listing.endTime);
382         _sendValue(originalBidder, originalAmount, _gasLimitMedium
383                 );
384     }
385 }
386 }
```

For the first bid, `if (listing.endTime == 0)` code block, the contract does not call `sendValue` nor `transferFrom(msgSender(), address(this), bidValue);`. This means that a user can place a bid as high as he wants and the contract would not check if the user actually owns those tokens.

If the amount bid is high enough, none else would be able to place a higher bid. This happens because after the first bid, the contract does require the user to send the tokens to the contract by calling `sendValue`

and `transferFrom` function as seen above.

Denial of Service example

First bid vs. Second bid

Risk Level:

Likelihood - 3

Impact - 5

Recommendation:

It is recommended to force the bidder of the first bid to transfer the amount into the contract following the same logic that the smart contract currently has now for all the bids that are not the first one. This can be achieved by simply adding these lines into the `if (listing.endTime == 0)`

code block:

Listing 2: Marketplace.sol - placeBid

```

1 if (currencyType == 2) {
2     IERC20(currencyContractAddress).transferFrom(msgSender(),
3         address(this), bidValue);
4
5 _sendValue(originalBidder, originalAmount, _gasLimitMedium);

```

Remediation Plan:

SOLVED: Rario team successfully fixed the `placeBid` function logic, correcting this vulnerability.

```

Calling --> rarioMarketplace.createAuction(1, 3000, "USDP", user1.address, {'from': 'user1'})
Transaction sent: 0x1c0d5295b9010545e14e070d081e0e0a70114714030d5c6
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 1
RarioMarketplace.createAuction confirmed Block: 13975143 Gas used: 401171 (5.97%)
token.ownerOf(1) --> 0xd97476316497970f0883db3d36934201d14e78a
user1.address --> 0x496e6aefcfa0d4f4238C719eF9f97AB79B9
rarioMarketplace.createAuction(1, 3000, "USDP", user1.address, {'from': 'user1'})
Gas price: 0.0 gwei Gas limit: 201010478a
rarioMarketplace.getListingIdByTokenId(1) --> 7854146079704491066829024063423341083692263798261161446442989268089806461
rarioMarketplace.getListingDetail(7854146079704491066829024063423341083692263798261161446442989268089806461, True) --> (3, 1, 3000, "USD", 0, '', '0x496e6aefcfa0d4f4238C719eF9f97AB79B9', '0x496e6aefcfa0d4f4238C719eF9f97AB79B9')
13975144 Gas used: 36502 (0.54%)
Calling --> rarioMarketplace.updateListingPrice confirmed Block: 13975144
Gas used: 36502 (0.54%)
Transaction sent: 0xb0befc699a5d4d94ea010dac3bd129dfe4aeeac5c13085704e24eb
Gas price: 0.0 gwei Gas limit: 201010478a
RarioMarketplace.updateListingPrice confirmed Block: 13975144 Gas used: 36502 (0.54%)
rarioMarketplace.getListingIdByTokenId(1) --> 7854146079704491066829024063423341083692263798261161446442989268089806461
rarioMarketplace.getListingDetail(7854146079704491066829024063423341083692263798261161446442989268089806461, True) --> (3, 1, 3000, "USD", 0, '', '0x496e6aefcfa0d4f4238C719eF9f97AB79B9', '0x496e6aefcfa0d4f4238C719eF9f97AB79B9')
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 29
00000000 -> 0x1994af1b9f6127417f0ff0ee529d3aaeb6d57a1e304a96e1d21d18357ea0c
Transaction sent: 0x1994af1b9f6127417f0ff0ee529d3aaeb6d57a1e304a96e1d21d18357ea0c
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 29
ERC20Token.transfer confirmed Block: 13975145 Gas used: 510K1 (0.76%)
Transaction sent: 0x05711117ed0d77274162f3d242370284a617532e919d931a7a0eac49507cb04
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 0
EthToToken.approve confirmed Block: 13975146 Gas used: 412t7 (0.64%)
user1.balanceOf(user1.address) --> 0
user1.balanceOf(user2.address) --> 0
usdc.balanceOf(user2.address) --> 49990000
usdc.balanceOf(rarioMarketplace.address) --> 0
Calling --> rarioMarketplace.placeBid(1, 3000, "USDC", user2.address, {'from': 'user2'})
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 1
RarioMarketplace.placeBid confirmed Block: 13975147 Gas used: 1830468 (2.72%)
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 1
RarioMarketplace.placeBid confirmed Block: 13975147 Gas used: 1830468 (2.72%)
token.ownerOf(1) --> 0xd97476316497970f0883db3d36934201d14e78a
user2.address --> 0x820e69a7b73dca19320a1086829024063423341083692263798261161446442989268089806461
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 29
00000000 -> 0x1994af1b9f6127417f0ff0ee529d3aaeb6d57a1e304a96e1d21d18357ea0c
Transaction sent: 0x1994af1b9f6127417f0ff0ee529d3aaeb6d57a1e304a96e1d21d18357ea0c
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 29
RarioMarketplace.placeBid confirmed Block: 13975148 Gas used: 49990002
Gas price: 0.0 gwei Gas limit: 4721978 Nounce: 29
user2.balanceOf(user1.address) --> 0
user2.balanceOf(treasury.address) --> 0
usdc.balanceOf(user1.address) --> 49990002
usdc.balanceOf(rarioMarketplace.address) --> 49990002
user2.balanceOf(1) --> 0xd97476316497970f0883db3d36934201d14e78a
user2.address --> 0x820e69a7b73dca19320a10868290240634236951d5b

```

3.2 (HAL-02) OUTBID USER DOES NOT RECEIVE BACK HIS FUNDS - HIGH

Description:

In the contract `Marketplace.sol`, the function `placeBid` is used to buy an NFT token listing. In case that the `listingType` is a `RESERVE_PRICE_AUCTION`, when a user places a bid, the amount of tokens bid are transferred from his account to the contract address. Then, when a user outbids the previous user, the new user token amount bid is also transferred to the contract but the previous outbid user is not sent his bid tokens back:

```
usdc._amount = usdcCurrencyManager.convert("USDC", "USDT", 6000) -> 59988002
Transaction sent: 0xfed70f1ed2edeadfb052f2ed0994a996e0e703489e24bdccb5cc99d61910912
Gas price: 0.0 gwei Gas limit: 0x219795Nonce: 0
ERC20Token.transfer confirmed Block: 13998949 Gas used: 51061 (0.76%)
Calling -> usdcMarketplace.placeBid(0x5a16607970449109682299024063423341036922637982611614464429926808906461, user3.address, "USDC", USDC_AMOUNT2 + 1, {'from': user3})
Gas price: 0.0 gwei Gas limit: 0x219795Nonce: 0
ERC20Token.approve confirmed Block: 13998950 Gas used: 44127 (0.66%)
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(treasury.address) -> 59988003
usdc._amount = usdcCurrencyManager.convert("USDC", "USDT", 6000)
Calling -> usdcMarketplace.placeBid(0x5a16607970449109682299024063423341036922637982611614464429926808906461) -> (2, 785616607970449109682299024063423341036922637982611614464429926808906461, 1, '0x54aF1a5A93743D421b79368936c008919c430
719368936c008919c430')
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(treasury.address) -> 59988003
usdc._amount = usdcCurrencyManager.convert("USDC", "USDT", 6000)
Calling -> usdcMarketplace.placeBid(0x5a16607970449109682299024063423341036922637982611614464429926808906461) -> (2, 785616607970449109682299024063423341036922637982611614464429926808906461, 1, '0x54aF1a5A93743D421b79368936c008919c430
0099175070HE17926466385773A4646236ew479', 6000, 2592000, 900, 1657935150)
usdc.balanceOf(usdcMarketplace.placeBid confirmed Block: 13998951 Gas used: 175500 (2.61%
usdcMarketplace.getListing(0x5a16607970449109682299024063423341036922637982611614464429926808906461) -> (2, 785616607970449109682299024063423341036922637982611614464429926808906461, 1, '0x54aF1a5A93743D421b79368936c008919c430
479368936c008919c430')
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(user3.address) -> 59988003
usdc._amount = usdcCurrencyManager.convert("USDC", "USDT", 7000)
Calling -> usdcMarketplace.placeBid(0x5a16607970449109682299024063423341036922637982611614464429926808906461, user3.address, "USDC", USDC_AMOUNT3 + 1, {'from': user3})
Gas price: 0.0 gwei Gas limit: 0x219795Nonce: 0
ERC20Token.transfer confirmed Block: 13998952 Gas used: 51061 (0.66%)
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(treasury.address) -> 0
usdc._amount = usdcCurrencyManager.convert("USDC", "USDT", 7000)
Calling -> usdcMarketplace.placeBid(0x5a16607970449109682299024063423341036922637982611614464429926808906461) -> (2, 785616607970449109682299024063423341036922637982611614464429926808906461, 1, '0x54aF1a5A93743D421b79368936c008919c430
0099175070HE17926466385773A4646236ew479', 7000, 2592000, 900, 1657935150)
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(user3.address) -> 0
usdc._amount = usdcCurrencyManager.convert("USDC", "USDT", 10000)
Calling -> usdcMarketplace.placeBid(0x5a16607970449109682299024063423341036922637982611614464429926808906461) -> (2, 785616607970449109682299024063423341036922637982611614464429926808906461, 1, '0x54aF1a5A93743D421b79368936c008919c430
129974006
token._amount(1) -> 0xb507b7FdeC75b0A8014231bA691b962383f5
user3._amount(1) -> 0
usdc.balanceOf(usdcMarketplace.placeBid confirmed Block: 13998954 Gas used: 149508 (2.28%
usdcMarketplace.getListing(0x5a16607970449109682299024063423341036922637982611614464429926808906461) -> (2, 785616607970449109682299024063423341036922637982611614464429926808906461, 1, '0x54aF1a5A93743D421b79368936c008919c430
479368936c008919c430')
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(treasury.address) -> 0
usdc._amount = usdcCurrencyManager.convert("USDC", "USDT", 10000)
Calling -> usdcMarketplace.placeBid(0x5a16607970449109682299024063423341036922637982611614464429926808906461) -> (2, 785616607970449109682299024063423341036922637982611614464429926808906461, 1, '0x54aF1a5A93743D421b79368936c008919c430
129974006
token._amount(1) -> 0xb507b7FdeC75b0A8014231bA691b962383f5
user3._amount(1) -> 0x532FD3994C0a19C7b08C5867b4985858410f00
```

It is worth mentioning that this issue does not occur when the payment is done with `NATIVE` currency. In this case, the funds are sent back to the outbid user. The issue only happens when the payment is done with `TOKENS`.

Risk Level:

Likelihood - 3

Impact - 5

Recommendation:

It is recommended that the smart contract sends back the bid tokens to the outbid user once a higher bid is placed.

Remediation Plan:

SOLVED: Rario team rightly modified the `placeBid` function and now refunds are given to the outbid users.

```
Calling -> rarioMarketplace.placeBid(785416607970449109688299024063432334108369226379826116161446442989268089806461, user2.address, "USDC", USDC_AMOUNT + 1, ('from': user2))
Transaction sent: 0xb0d7a5f759c5...024063432334108369226379826116161446442989268089806461
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
RarioMarketplace.placeBid confirmed Block: 13975147 Gas used: 189068 (2.72%)
rarioMarketplace.getListingDetail(785416607970449109688299024063432334108369226379826116161446442989268089806461, True) -> (3, 1, 5000, "USD", 49980002, "USDC", "0x49eadaedfca66fa23bc719af9f", "0x49eadaedfca66fa23bc719af9f")
DE7ABF759c5..., 2552000
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(user2.address) -> 0
usdc.balanceOf(user3.address) -> 0
usdc.balanceOf(rarioMarketplace.address) -> 4550002
user1.address -> 0x6e20d9a8773cd1a9320a74f23945c0c05bf28
user2.address -> 0x6e20d9a8773cd1a9320a74f23945c0c05bf28
USDC_AMOUNT2 = rarioCurrencyManager.convert("USD", "USDC", 5000) -> 5998002
Transaction sent: 0xb0d7a5f759c5...024063432334108369226379826116161446442989268089806461
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 3
ERC20Token.transfer confirmed Block: 13975148 Gas used: 31061 (0.76%)
Transaction sent: 0xb0d7a5f759c5...024063432334108369226379826116161446442989268089806461
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 0
ERC20Token.approve confirmed Block: 13975149 Gas used: 44127 (0.66%)
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(treasury.address) -> 0
usdc.balanceOf(user2.address) -> 0
usdc.balanceOf(user3.address) -> 5998003
usdc.balanceOf(rarioMarketplace.address) -> 4550002
Call -> rarioMarketplace.getListingDetail(785416607970449109688299024063432334108369226379826116161446442989268089806461, user3.address, "USDC", USDC_AMOUNT3 + 1, ('from': user3))
Transaction sent: 0xb0d7a5f759c5...024063432334108369226379826116161446442989268089806461
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
RarioMarketplace.placeBid confirmed Block: 13975150 Gas used: 169841 (1.53%)
rarioMarketplace.getListingDetail(785416607970449109688299024063432334108369226379826116161446442989268089806461, True) -> (3, 1, 6000, "USD", 5998003, "USDC", "0x49eadaedfca66fa23bc719af9f", "0x49eadaedfca66fa23bc719af9f")
DE7ABF759c5..., 2552000
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(treasury.address) -> 0
usdc.balanceOf(user2.address) -> 4550002
usdc.balanceOf(user3.address) -> 5998003
usdc.balanceOf(rarioMarketplace.address) -> 5998003
user1.address -> 0xb767a3e6d47cc64d165c1354c58b9c987c98
USDC_AMOUNT3 = rarioCurrencyManager.convert("USD", "USDC", 7000) -> 69986002
```

3.3 (HAL-03) UNCHECKED TRANSFER - MEDIUM

Description:

In the contract `Marketplace.sol` the return value of some external transfer/transferFrom calls are not checked. Several tokens do not revert in case of failure and return false. If one of these tokens is used, a deposit would not revert if the transfer fails, and an attacker could deposit tokens for free.

Code Location:

- Line 370: `IERC20(currencyContractAddress).transferFrom(msgSender(), address(this), bidValue);`
- Line 611: `IERC20(contractAddress).transferFrom(msgSender(), _treasury, platformPayableAmount);`
- Line 612: `IERC20(contractAddress).transferFrom(msgSender(), seller, sellerPayableAmount);`

Risk Level:

Likelihood - 2

Impact - 4

Recommendation:

It is recommended to use `SafeERC20`.

Remediation Plan:

SOLVED: Rario team uses now `SafeERC20` in all the transfers.

3.4 (HAL-04) USE OF TRANSFER WHEN TRANSFERRING NFTS - LOW

Description:

In the contract `Marketplace.sol` the function `IERC721(_trustedToken).transferFrom` is called when transferring the NFT:

- Line 351: `IERC721(_trustedToken).transferFrom(address(this), sellerTokenWallet, tokenId);`
- Line 539: `IERC721(_trustedToken).transferFrom(_msgSender(), address(this), tokenId);`
- Line 622: `IERC721(_trustedToken).transferFrom(address(this), listing.buyerTokenWallet, listing.tokenId);`

However, this function does not check whether the recipient is aware of the ERC721 protocol and calls `_transfer` directly. If the recipient is a contract not aware of incoming NFTs, then the transferred NFT would be locked in the recipient forever.

Risk Level:

Likelihood - 2

Impact - 2

Recommendation:

It is recommended to use `SafeTransferFrom` which internally calls `_safeTransfer` checking if the recipient contract implements the `onERC721Received` interface to avoid loss of NFTs.

Remediation Plan:

SOLVED: Rario team uses `IERC721.safeTransferFrom` when sending the token from the smart contract to any user/wallet:

```
IERC721(_trustedToken).safeTransferFrom(address(this), listing.buyerTokenWallet, listing tokenId);
```

At the same time, to avoid implementing the `onERC721Received` function, Rario team has decided to keep using `IERC721.transfer` when transferring tokens from any user to their smart contract address:

```
IERC721(_trustedToken).transferFrom(_msgSender(), address(this), tokenId);
```

3.5 (HAL-05) UPDATEUSER FUNCTION MISSING REQUIRE STATEMENT - LOW

Description:

In the contract `UserDB.sol` the function `addUser` checks that the wallet address has not been added previously:

Listing 3: UserDB.sol - addUser (Lines 45)

```
41 function addUser(uint256 rarioId, address walletAddress) external
    override {
42     _checkRole(OPERATOR, msgSender());
43     require(walletAddress != address(0), ERR_ZERO_ADDRESS_WALLET);
44     require(!rarioIds.contains(rarioId), ERR_USER_ALREADY_EXISTS);
45     require(walletAddressToRarioId[walletAddress] == 0,
        ERR_WALLET_ALREADY_EXISTS);
46
47     rarioIds.add(rarioId);
48     rarioIdToWalletAddress[rarioId] = walletAddress;
49     walletAddressToRarioId[walletAddress] = rarioId;
50     emit UserAdded(rarioId, walletAddress);
51 }
```

The function `updateUser` allows to update that address but is missing that validation:

Listing 4: UserDB.sol - updateUser

```
140 function updateUser(uint256 rarioId, address walletAddress)
    external override {
141     _checkRole(OPERATOR, msgSender());
142     require(walletAddress != address(0), ERR_ZERO_ADDRESS_WALLET);
143     require(rarioIds.contains(rarioId), ERR_USER_DOES_NOT_EXIST);
144
145     address oldAddress = rarioIdToWalletAddress[rarioId];
146
147     rarioIdToWalletAddress[rarioId] = walletAddress;
148     walletAddressToRarioId[walletAddress] = rarioId;
149     delete walletAddressToRarioId[oldAddress];
150 }
```

```

151     emit UserUpdated(rarioId, oldAddress, walletAddress);
152 }

```

This can cause some inconsistencies, for example, with the function `getUserByAddress`:

```

>>> rariousers getUserById(1
(), "0x1B1c0E70CFCfd1d1b25c667a48bF010b7E5196C3")
>>> rariousers getUserById(2
(), "0x3BDE98b2F0f540b38de46ca679bb288E01D0D61B")
>>> rariousers updateUser(rarioId, user.address)
Transaction submitted successfully: 0x5fb01abf3caldbbb4c45bb1b5d13df1216b5053f
Gas price: 0.0 gwei Gas limit: 4721975 Nonce: 29
RarioUsers.updateUser confirmed Block: 13511179 Gas used: 29009 (0.43%)
<Transaction '0x5fb01abf3caldbbb4c45bb1b5d13df1216b5053f'>
>>> rariousers getUserById(1
(), "0x1B1c0E70CFCfd1d1b25c667a48bF010b7E5196C3")
>>> rariousers getUserById(2
(), "0x1B1c0E70CFCfd1d1b25c667a48bF010b7E5196C3")
>>> rariousers getUserByAddress(user.address)
(), "0x1B1c0E70CFCfd1d1b25c667a48bF010b7E5196C3")

```

Risk Level:

Likelihood - 1

Impact - 3

Recommendation:

It is recommended to add the missing require statement to the `updateUser` function.

Remediation Plan:

SOLVED: Rario team added the missing require statement:

Listing 5: UserDB.sol – updateUser (Lines 144)

```

140 function updateUser(uint256 rarioId, address walletAddress)
    external override {
141     _checkRole(OPERATOR, _msgSender());
142     require(walletAddress != address(0), ERR_ZERO_ADDRESS_WALLET);
143     require(rarioIds.contains(rarioId), ERR_USER_DOES_NOT_EXIST);
144     require(walletAddressToRarioId[walletAddress] == 0,
              ERR_WALLET_ALREADY_EXISTS);
145
146     address oldAddress = rarioIdToWalletAddress[rarioId];

```

FINDINGS & TECH DETAILS

```
147
148     rarioIdToWalletAddress[rarioId] = walletAddress;
149     walletAddressToRarioId[walletAddress] = rarioId;
150     delete walletAddressToRarioId[oldAddress];
151
152     emit UserUpdated(rarioId, oldAddress, walletAddress);
153 }
```

3.6 (HAL-06) MINT FUNCTION IN RARIOPOLYGONNFT DOES NOT VERIFY IF TOKEN WAS WITHDRAWN - INFORMATIONAL

Description:

The contract `RarioPolygonNFT`, as per the [documentation](#), contains the functions `deposit`, `withdraw` and `mint`. The `mint` function is inherited from `RarioNFT` and contains the following code:

Listing 6: RarioNFT.sol - mint

```
39 function mint(address to, uint256 tokenId) external virtual {
40     _checkRole(OPERATOR, msgSender());
41     _mint(to, tokenId);
42 }
```

The function does not verify if the token was withdrawn previously as it is done in `ChildMintableERC721.sol`:

Listing 7: ChildMintableERC721.sol - mint (Lines 2268,2273)

```
2264 /**
2265 * @notice Example function to handle minting tokens on matic
2266 * chain
2267 * @dev Minting can be done as per requirement,
2268 * This implementation allows only admin to mint tokens but it can
2269 * be changed as per requirement
2270 * Should verify if token is withdrawn by checking `withdrawnTokens` mapping
2271 */
2272 function mint(address user, uint256 tokenId) public only(
2273     DEFAULT_ADMIN_ROLE) {
        require(!withdrawnTokens[tokenId], "ChildMintableERC721:
TOKEN_EXISTS_ON_ROOT_CHAIN");
```

```
2274     _mint(user, tokenId);
2275 }
```

Risk Level:

Likelihood - 1

Impact - 1

Recommendation:

It is recommended to check in `RarioPolygonNFT` if the token was withdrawn before minting.

Remediation Plan:

SOLVED: `Rario team` modified the `mint` function in `RarioPolygonNFT`. Now the function checks if the token was withdrawn before minting:

Listing 8: RarioPolygonNFT.sol - mint (Lines 34)

```
32 function mint(address user, uint256 tokenId) external override {
33     _checkRole(OPTIONAL, _msgSender());
34     require(!withdrawnTokens[tokenId], "ChildMintableERC721:
35         TOKEN_EXISTS_ON_ROOT_CHAIN");
36     _mint(user, tokenId);
37 }
```

3.7 (HAL-07) POSSIBLE MISUSE OF PUBLIC FUNCTIONS - INFORMATIONAL

Description:

In the contract `Marketplace.sol` there are functions marked as public but they are never directly called within the same contract or in any of its descendants:

`Marketplace.sol`

- `getPendingWithdrawal(address)` (`Marketplace.sol#387-389`)
- `withdraw()` (`Marketplace.sol#394-396`)

Risk Level:

Likelihood - 1

Impact - 1

Recommendation:

If the function is not intended to be called internally or by descendants, it is better to mark all these functions as `external` to save gas.

Remediation Plan:

SOLVED: Rario team declared those functions as external.

3.8 (HAL-08) UNUSED CONSTANTS - INFORMATIONAL

Description:

The following constants are not used anywhere in the code.

Code Location:

Listing 9: ERC721NFT.sol

```
30 string internal constant E_NFT_TRANSFER_TO_NON_RECEIVER = "E03001"
      ;
```

Listing 10: Marketplace.sol

```
57 string private constant ERR_CURRENCY_MANAGER = "NFTMarketplace:
      Error in CurrencyManager";
```

Risk Level:

Likelihood - 1

Impact - 1

Recommendation:

Check whether these constants should be used and if not remove them in order to save gas.

Remediation Plan:

SOLVED: Rario team removed these unused constants.

3.9 (HAL-09) UNUSED EVENTS - INFORMATIONAL

Description:

The following events are declared but they are not emitted by any function:

Listing 11: MetadataDB.sol

```
105 event TokenStringValueUpdated(uint256 indexed tokenId, string
      indexed attribute, string value);
106 event TokenNumberValueUpdated(uint256 indexed tokenId, string
      indexed attribute, int256 value);
107 event TokenBooleanValueUpdated(uint256 indexed tokenId, string
      indexed attribute, bool value);
```

Listing 12: Marketplace.sol

```
140 event WithdrawPending(address indexed user, uint256 amount);
```

Listing 13: CurrencyManager.sol

```
97 event BaseCurrencyUpdated(string indexed symbol);
```

Listing 14: MarketplaceConfig.sol

```
49 event TokenPriceFeedUpdated(address indexed newAddress, address
      indexed oldAddress, string symbol);
```

Risk Level:

Likelihood - 1

Impact - 1

FINDINGS & TECH DETAILS

Recommendation:

Check whether these events should be used and if not remove them.

Remediation Plan:

SOLVED: Rario team removed all the unused events from their smart contracts.

MANUAL TESTING

4.1 INTRODUCTION

Halborn performed different manual tests in all the contracts trying to find logic flaws and vulnerabilities that were not detected by the automatic tools.

During the manual testing multiple questions were considered while evaluating each of the defined functions:

- Can it be re-called changing admin/roles and permissions?
- Can somehow an external controlled contract call again the function during the execution of it? (Re-entrancy)
- Can a function be called twice in the same block causing issues?
- Do we control sensitive or vulnerable parameters?
- Does the function check for boundaries on the parameters and internal values? Bigger than zero or equal? Argument count, array sizes, integer truncation. . .
- Are the function parameters and variables controlled by external contracts?
- Can extended contracts cause issues on the extender contract?

4.2 UPGRADEABLE CONTRACTS' ARCHITECTURE

As our first test, we validated that the upgradeable contracts' architecture was correct. This is the hierarchy of the contract `Upgradeable`:

1. `ERC1967Upgrade`: This abstract contract provides getters and event emitting update functions for EIP1967 slots.
2. `ERC1822UUPS`: Inherits from `ERC1967Upgrade`. This contract is used as an upgradeability mechanism designed for UUPS proxies. Correctly initializes `ERC1967Upgrade`:

Listing 15: ERC1822UUPS.sol - initializeERC1822UUPS (Lines 22)

```
21 function initializeERC1822UUPS() internal initializer {
22     initializeERC1967Upgrade();
23 }
```

3. `Upgradeable`: Inherits from `ERC1822UUPS`, `Constants`, `Initializable`, `AccessRoles` and `Pausable`. Correctly initializes all of its parent contracts:

Listing 16: Upgradeable.sol - initializeUpgradeable

```
16 function initializeUpgradeable(address trustedForwarder) internal
17     initializer {
18     setTrustedForwarder(trustedForwarder);
19     initializeAccessControl();
20     initializePausable();
21     initializeERC1822UUPS();
22     _setupRole(DEFAULT_ADMIN_ROLE, msgSender());
23     _setupRole(SUPER_RARIO, msgSender());
24     _setupRole(RARIO_ADMIN, msgSender());
25     _setupRole(OPERATOR, msgSender());
26 }
```

Moreover, it overrides `ERC1822UUPS._authorizeUpgrade` function, adding the modifier `whenPaused` and checking that `msgSender()` has `SUPER_RARIO` role. This way the contract will only be able to be upgraded by members of the `SUPER_RARIO` group and only if the contract is paused.

Listing 17: Upgradeable.sol - authorizeUpgrade

```
43 function _authorizeUpgrade(address) internal view override
44     whenPaused {
45         _checkRole(SUPER_RARIO, msgSender());
46 }
```

4.3 CONTRACTS INITIALIZATION

No issues found on the initialization. Although:

1. CurrencyManager, ERC721NFT, MarketplaceConfig and RarioEthereumNFT init functions should be kept as internal as they are missing the initializer modifier.
2. We recommend calling `initializeReentrancyGuard()` in the contract Marketplace `initializeMarketplace` function.

Contract	Inherits from	Has initialize function?	Has initializer modifier?	Code
RarioCurrencyManager	CurrencyManager	Yes	Yes	<pre> 20 function initialize(address trustedForwarder) external initializer { 21 initializeUpgradable(trustedForwarder); 22 } </pre>
RarioData	MetadataDB	Yes, MetadataDB has no initialize function, hence it is not initialized in RarioData	Yes	<pre> 20 function initialize(address trustedForwarder) external initializer { 21 initializeUpgradable(trustedForwarder); 22 } </pre>
RarioEthereumToken	RarioEthereumNFT	Yes	Yes	<pre> 14 function initialize(15 address trustedForwarder_, 16 address rarioData_, 17 address rarioUserDB_, 18 string calldata name_, 19 string calldata symbol_ 20) external initializer { 21 initializeUpgradable(trustedForwarder_); 22 initializeRarioEthereumNFT(name_, symbol_, rarioData_, rootChainManagerAddress_); 23 </pre>
RarioMarketplace	Marketplace	Yes	Yes	<pre> 37 function initialize(address trustedForwarder, address configManager, address currencyManager) external initializer { 38 initializeUpgradable(trustedForwarder); 39 initializeMarketplace(configManager, currencyManager); 40 41 name = "RarioMarketplace"; 42 version = "3"; 43 </pre>
RarioMarketplaceConfig	MarketplaceConfig	Yes	Yes	<pre> 34 function initialize(35 address trustedForwarder, 36 address payable treasury, 37 address tokenContract, 38 address rarioUserDB_ 39) external initializer { 40 initializeUpgradable(trustedForwarder); 41 initializeMarketplaceConfig(treasury, tokenContract, rarioUserDB_); 42 </pre>
RarioPolygonToken	RarioPolygonNFT	Yes	Yes	<pre> 14 function initialize(15 address trustedForwarder_, 16 address rarioData_, 17 address predicateProxy_, 18 string calldata name_, 19 string calldata symbol_ 20) external initializer { 21 initializeUpgradable(trustedForwarder_); 22 initializeRarioPolygonNFT(name_, symbol_, rarioData_, predicateProxy_); 23 </pre>
RarioUsers	UserDB	Yes, UserDB has no initialize function, hence it is not initialized in RarioUsers	Yes	<pre> 20 function initialize(address trustedForwarder) external initializer { 21 initializeUpgradable(trustedForwarder); 22 </pre>
AccessControl	Initializable, Context, IAccessControl, IAccessControlEnumerable, ERC165Interface	Yes, internal function. It is called in the Upgradable contract	Yes	<pre> 56 function initializeAccessControl() internal initializer { 57 registerInterface(type(IAccessControl).interfaceId); 58 registerInterface(type(IAccessControlEnumerable).interfaceId); 59 </pre>
Context				
CurrencyManager	Upgradeable, ICurrencyManager	No	Yes	<p>No, although it is internal and is called by RarioCurrencyManager init function, it is correctly protected with the initializer modifier, hence it is safe as long as it is kept as an internal function</p> <pre> 100 function initializeCurrencyManager(string calldata nativeCurrencySymbol) internal { 101 _setNativeDecimals(2); 102 _addCurrency(CurrencyType.FIAT, "USD", address(0), _fiatDecimals); 103 _nativeCurrency = nativeCurrencySymbol; 104 </pre>
ERC165Interface	Initializable, IERC165	Yes	Yes	<pre> 20 function initializeERC165Interface() internal initializer { 21 registerInterface(type(IERC165).interfaceId); 22 </pre>
ERC721NFT	Upgradeable, IERC721, IERC721Metadata, IERC721Enumerable	Yes	No	<p>No, although it is internal and is called by RarioNFT init function, it is correctly protected with the initializer modifier, hence it is safe as long as it is kept as an internal function</p> <pre> 56 function initializeERC721NFT(string memory name_, string memory symbol_) internal { 57 require(bytes(name_).length > 0, E_NFT_EMPTY_TOKEN_NAME); 58 require(bytes(symbol_).length > 0, E_NFT_EMPTY_TOKEN_SYMBOL); 59 60 name = name_; 61 symbol = symbol_; 62 63 initializeERC165Interface(); 64 registerInterface(type(IERC721).interfaceId); 65 registerInterface(type(IERC721Metadata).interfaceId); 66 registerInterface(type(IERC721Enumerable).interfaceId); 67 68 69 70 71 72 73 74 75 76 77 </pre>
ERC1822UUPS	Initializable, ERC1967Upgrade	Yes	Yes	<pre> 21 function initializeERC1822UUPS() internal initializer { 22 initializeERC1967Upgrade(); 23 </pre>
ERC1967Upgrade	Initializable	Yes	Yes	<pre> 39 function initializeERC1967Upgrade() internal initializer {} </pre>
InitializableMarketplace	Upgradeable, ReentrancyGuard	No	Yes	<p>Yes, although ReentrancyGuard is not initialized by calling <code>initializeReentrancyGuard()</code>, it is recommended to initialize the ReentrancyGuard contract separately if the current code has no security impact</p> <pre> 143 function initializeMarketplace(address config, address currencyManager) internal initializer { 144 config = IMarketplaceConfig(config); 145 _currencyManager = ICurrencyManager(currencyManager); 146 _hasConfiguration = false; 147 </pre>

MarketplaceConfig	Upgradable, MarketplaceConfig	Yes	No, although it is internal and is called by RenoMarketplaceConfig init function which is correctly protected with the initializer modifier, hence it is safe as long as it is kept as an internal function	<pre> 65 function initializeMarketplaceConfig(66 address payable treasury, 67 address tokenContract, 68 address rarioUserDB 69) internal { 70 _setTreasury(treasury); 71 _setTokenContract(tokenContract); 72 _setRarioUserDB(rarioUserDB); 73 _setGasLimits(200000, 120000, 20000); 74 _setPlatformFee(0.0001); 75 _setAllowExternalMalleability(false); 76 _setListingExpiry(30); 77 _setAuctionExtensionDuration(15); 78 _setAuctionPercentIncrementInBasisPoints(500); 79 } </pre>
MetadataDB	Upgradable, IMetadataDB	No, although it is worth mentioning that although Upgradable is called and protected by all the contracts that inherits from MetadataDB, for example, RenoDB	-	-
Pausable	Initializable, Context	No, internal function. It is called in the Upgradable contract	Yes	<pre> 30 function initializePausable() internal initializer { 31 _paused = false; 32 } </pre>
RarioEthereumNFT	IMutableERC721, RarioNFT	Yes, internal function. It is called in the RarioEthereumToken contract	No, although it is internal and is called in the RarioEthereumToken contract	<pre> 12 function initializeRarioEthereumNFT(string memory name_, string memory symbol_, address rarioData_, address rootChainManagerAddress_) { 13 initializeRarioNFT(name_, symbol_, rarioData_); 14 _setupRole(PREDICATE_ROLE, msgSender()); 15 _setupRole(PREDICATE_ROLE, rootChainManagerAddress_); 16 } </pre>
RarioNFT	ERC721NFT	Yes	Yes	<pre> 19 function initializeRarioNFT(20 string memory name_, 21 string memory symbol_, 22 address rarioData_ 23) internal initializer { 24 initializeERC721NFT(name_, symbol_); 25 _rarioData = IMetadataDB(rarioData_); 26 } </pre>
RarioPolygonNFT	IPolyonChildToken, RarioNFT	Yes, internal function. It is called in the RarioPolygonToken contract	Yes	<pre> 19 function initializeRarioPolygonNFT(string memory name_, string memory symbol_, address rarioData_, address predicateProxy_) internal initializer { 20 initializeRarioNFT(name_, symbol_, rarioData_); 21 _setupRole(DEPOSITOR_ROLE, predicateProxy_); 22 } </pre>
ReentrancyGuard	Initializable	Yes	Yes	<pre> 39 function initializeReentrancyGuard() internal initializer { 40 _status = NOT_ENTERED; 41 } </pre>
Upgradable	Context, AccessControl, Pausable, ERC1822UIUPS	Yes, internal function. It is called in all the contracts that inherits from this one	Yes	<pre> 16 function initializeUpgradable(address trustedForwarder) internal initializer { 17 setTrustedForwarder(trustedForwarder); 18 initializeAccessControl(); 19 initializePausable(); 20 initializeERC1822UIUPS(); 21 _setupRole(DEFAULT_ADMIN_ROLE, msgSender()); 22 _setupRole(SUPER_RARIO, msgSender()); 23 _setupRole(RARIO_ADMIN, msgSender()); 24 _setupRole(OPTIONATOR, msgSender()); 25 } 26 } </pre>
UserDB	Upgradable, IUserDB	No, although it is worth mentioning that although Upgradable is called and protected by all the contracts that inherits from UserDB, for example, RarioUsers	-	-

4.4 RARIOCURRENCYMANAGER CONTRACT

The contract `RarioCurrencyManager` inherits from `CurrencyManager`. This contract is used to store all the different attributes related to the tokens.

The contract contains the following functions:

- From `CurrencyManager`
 - `_addCurrency(CurrencyManager.CurrencyType, string, address, uint8)` (private)
 - `_convert(string, string, uint256)` (private)
 - `_getCurrency(string)` (private)
 - `_getCurrencyPair(string)` (private)
 - `_memcmp(bytes, bytes)` (internal)
 - `_removeCurrency(string)` (private)
 - `_removeCurrencyPair(string)` (private)
 - `_setFiatDecimals(uint8)` (private)
 - `_strcmp(string, string)` (internal)
 - `addCryptoCurrency(string, address, uint8)` (external)
 - `addCurrencyPair(string, string, string, bool, uint8, address)` (external)
 - `addFiatCurrency(string)` (external)
 - `convert(string, string, uint256)` (external)
 - `getCurrency(string)` (external)
 - `getCurrencyByIndex(uint256)` (external)
 - `getCurrencyPair(string)` (external)
 - `getCurrencyPairByIndex(uint256)` (external)
 - `getFiatDecimals()` (external)
 - `getNativeCurrency()` (external)
 - `initializeCurrencyManager(string)` (internal)
 - `removeCurrency(string)` (external)
 - `removeCurrencyPair(string)` (external)
 - `setFiatDecimals(uint8)` (external)
 - `supportedCurrencies()` (external)
 - `supportedCurrencyPairs()` (external)

- `supportsCurrency(string)` (external)
- `supportsCurrencyPair(string)` (external)
- `totalSupportedCurrencies()` (external)
- `totalSupportedCurrencyPairs()` (external)
- From `RarioCurrencyManager`
- `initialize(address, string)` (external)

As this contract was really simple our manual tests focused on checking that all the getter and setter functions were working as expected:

```
Contract RarioCurrencyManager (rarioCurrencyManager) deployed at 0x1AF81ED448b89aa0C5224051dbe6ED02f471Cd2B
Transaction sent: 0xecd6e368c5fd9101a7d2de8e62ad7df289b7fd32eeb3e0211e984de97
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 4
RarioCurrencyManager.initialize confirmed Block: 13492144 Gas used: 620118 (9.23%)
Contract RarioCurrencyManager initialized

Calling -> rarioCurrencyManager.addFiatCurrency("EURO", {'from': owner})
Transaction sent: 0xa0072337fd8cfed0d65513d2d3hca5b990ffle30e8040266448ca0bbbe75a9
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 5
RarioCurrencyManager.addFiatCurrency confirmed Block: 13492145 Gas used: 123829 (1.84%)

Calling -> rarioCurrencyManager.addFiatCurrency("DOLLAR", {'from': owner})
Transaction sent: 0x8fa1a8584770555511668d17752a63e3e02551e93750cf8163920
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 6
RarioCurrencyManager.addFiatCurrency confirmed Block: 13492146 Gas used: 123893 (1.84%)

rarioCurrencyManager.getNativeCurrency() -> MATIC

Calling -> rarioCurrencyManager.addCryptoCurrency("MATIC", user0.address, 18), {'from': owner})
Transaction sent: 0x8fa1a8584770555511668d17752a63e3e02551e93750cf8163920
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 7
RarioCurrencyManager.addCryptoCurrency confirmed Block: 13492147 Gas used: 146210 (2.18%)

Calling -> rarioCurrencyManager.addCurrencyPair("MATICUSD", "MATIC", "USD", False, 8, maticFeed.address, {'from': owner})
Transaction sent: 0x509cb00a735193096d7d7e40d2e0accbed7e1959bef92db4a5fcfe724
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 8
RarioCurrencyManager.addCurrencyPair confirmed Block: 13492148 Gas used: 184994 (2.75%)

rarioCurrencyManager.getFiatDecimals() -> 2
Calling -> rarioCurrencyManager.setFiatDecimals(5, {'from': owner})
Transaction sent: 0xd535a953b00a9c0247aabe97a0e09788e91c7a7f6c89b6c9a44a37b9ba
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 9
RarioCurrencyManager.setFiatDecimals confirmed Block: 13492149 Gas used: 73443 (1.09%)

rarioCurrencyManager.getFiatDecimals() -> 5
rarioCurrencyManager.getCurrency("EUR0") -> ('EURO', 0, '0x0000000000000000000000000000000000000000000000000000000000000000', 5)
rarioCurrencyManager.getCurrency("DOLLAR") -> ('DOLLAR', 0, '0x0000000000000000000000000000000000000000000000000000000000000000', 5)
rarioCurrencyManager.getCurrency("MATIC") -> ('MATIC', 1, '0x0000000000000000000000000000000000000000000000000000000000000000', 18)
rarioCurrencyManager.getCurrencyByIndex(1) -> ('USD', 0, '0x0000000000000000000000000000000000000000000000000000000000000000', 5)
rarioCurrencyManager.getCurrencyByIndex(1) -> ('EURO', 0, '0x0000000000000000000000000000000000000000000000000000000000000000', 5)
rarioCurrencyManager.getCurrencyByIndex(1) -> ('DOLLAR', 0, '0x0000000000000000000000000000000000000000000000000000000000000000', 5)
rarioCurrencyManager.getCurrencyByIndex(1) -> ('MATIC', 1, '0x0000000000000000000000000000000000000000000000000000000000000000', 18)
rarioCurrencyManager.getSupportedCurrencies() -> ('MATICUSD', 'MATIC', 'USD', False, 8, '0x67C9b67329c684f4E03fb4C7cc3C09183C7141dE')
maticFeed.address -> 0x67C9b67329c684f4E03fb4C7cc3C09183C7141dE
rarioCurrencyManager.getCurrencyPairByIndex(0) -> ('MATICUSD', 'MATIC', 'USD', False, 8, '0x67C9b67329c684f4E03fb4C7cc3C09183C7141dE')
rarioCurrencyManager.supportedCurrencies() -> ('USD', 'EURO', 'DOLLAR', 'MATIC')
rarioCurrencyManager.supportsCurrency('PAIR') -> True
rarioCurrencyManager.supportsCurrency('EUR0') -> True
rarioCurrencyManager.supportsCurrency('PAIR') -> True
rarioCurrencyManager.totalSupportedCurrencies() -> 4
rarioCurrencyManager.totalSupportedCurrencyPairs() -> 1
```

The setter functions can only be called by `RARIO_ADMIN`. The getter functions can be called by anyone, except `getFiatDecimals()` which can only be called by `OPERATORS`.

4.5 RARIODATA CONTRACT

The contract `RarioData` inherits from `MetadataDB`. This contract is used to store all the different attributes related to the tokens.

The contract contains the following functions:

- From `MetadataDB`
 - `_canSetValue(uint256, string)` (private)
 - `_getAttribute(string)` (private)
 - `addAttribute(string, string, MetadataDB.AttributeType, bool)` (external)
 - `depricate(string)` (external)
 - `existsAttribute(string)` (external)
 - `getAttribute(string)` (external)
 - `getAttributeByIndex(uint256)` (external)
 - `getAttributeCount()` (external)
 - `getBooleanValue(uint256, string)` (external)
 - `getNumberValue(uint256, string)` (external)
 - `getTextValue(uint256, string)` (external)
 - `setBooleanValue(uint256, string, bool)` (external)
 - `setDescription(string, string)` (external)
 - `setNumberValue(uint256, string, int256)` (external)
 - `setTextValue(uint256, string, string)` (external)
- From `RarioData`
 - `initialize(address)` (external)

All the setter functions in this contract can only be called by users with the `OPERATOR` role except the function `existsAttribute(string)` which can be called by anyone. On the other hand, the setter functions can only be called by users with the `RARIO_ADMIN` role.

We can see below how the different getter and setter functions are working as expected and also how they are restricted with the right access control policies:

```

Calling -> rariodata.addAttribute("label0", "description0", 0, False, {'from': owner})
Transaction sent: 0x6dded9d0cde1e9cb5b05aee4d8389291272811298b39elaeec0e0a7e
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 2
RarioData.addAttribute confirmed Block: 13492598 Gas used: 122780 (1.83)

Calling -> rariodata.addAttribute("label1", "description1", 1, True, {'from': owner})
Transaction sent: 0xf65e680c2d9462d67ff54547043a78cfefc39dd483835573lfaea842e9e86284
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 3
RarioData.addAttribute confirmed Block: 13492590 Gas used: 146204 (2.18%)

Calling -> rariodata.addAttribute("label2", "description2", 2, True, {'from': owner})
Transaction sent: 0x950f073cc8e854ea3fc01046shsbfrb0012fdrc7cl9db9efaf8e0b7be9556014
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 4
RarioData.addAttribute confirmed Block: 13492591 Gas used: 146204 (2.18%)

rariodata.existsAttribute("label0") -> True
rariodata.existsAttribute("label1") -> True
rariodata.existsAttribute("label2") -> True
rariodata.getAttribute("label0", {'from': owner}) -> {'label0', 'description0', 'text', False, False}
rariodata.getAttribute("label1", {'from': owner}) -> {'label1', 'description1', 'boolean', True, True}
rariodata.getAttribute("label2", {'from': owner}) -> {'label2', 'description2', 'number', True, False}
rariodata.getAttributeByIndex(0, {'from': owner}) -> {'label0', 'description0', 'text', False, False}
rariodata.getAttributeByIndex(1, {'from': owner}) -> {'label1', 'description1', 'boolean', True, True}
rariodata.getAttributeByIndex(2, {'from': owner}) -> {'label2', 'description2', 'number', True, False}
rariodata.deprecate confirmed Block: 13492592 Gas used: 36481 (0.54%)
Calling -> rariodata.deprecate("label1", {"from": owner})
Transaction sent: 0x9515c1c545056dmc8f5c7e65d0b5e10b43a4537a22d71bdca7162b87d2e9a91
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 5
RarioData.deprecate confirmed Block: 13492592 Gas used: 36481 (0.54%)

rariodata.getAttribute("label0", {'from': owner}) -> {'label0', 'description0', 'text', False, False}
rariodata.getAttribute("label0", {'from': owner}) -> {'label1', 'description1', 'boolean', True, True}
rariodata.getAttribute("label0", {'from': owner}) -> {'label2', 'description2', 'number', True, False}
rariodata.getAttribute("label1", {"from": owner}) -> False
Calling -> rarioData.setBooleanValue(1, "label1", True, {"from": owner})
Transaction sent: 0x1c1a1708a0940467aa215358hd304be4f4cbf527b0946c7a3790288c0baef
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 6
RarioData.setBooleanValue confirmed Block: 13492593 Gas used: 152308 (2.27%)

rariodata.getBooleanValue(1, "label1") -> True
Calling -> rarioData.setBooleanValue(1, "label0", False, {"from": owner})
Transaction sent: 0x216cf7e8c708613191c1d5a87090ah52idba58d38aa315994214ebdb8b8830
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 7
RarioData.setBooleanValue confirmed Block: 13492594 Gas used: 40326 (0.60%)

rariodata.getBooleanValue(1, "label0") -> False
Calling -> rarioData.setDescription("label0", "NEWDESCRIPTION", {"from": owner})
Transaction sent: 0x3a34ee98ab2cc933d58d2b56cc6c1454547803feacec799b7d667de7a226d
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 8
RarioData.setDescription confirmed Block: 13492595 Gas used: 42480 (0.63%)
rariodata.getAttribute("label0", {'from': owner}) -> {'label0', 'NEWDESCRIPTION', 'text', False, False}
rariodata.getAttribute("label0", {'from': owner}) -> {'label1', 'description1', 'boolean', True, True}
rariodata.getAttribute("label0", {'from': owner}) -> {'label2', 'description2', 'number', True, False}
Calling -> rarioData.setNumberValue(1, "label0", 1337, {"from": owner})
Transaction sent: 0x1d45dcf3833bea3c9c82cf7042518aeddffdd7ead425cb23a3c031c5edb
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 9
RarioData.setNumberValue confirmed Block: 13492596 Gas used: 69458 (1.03%)
rariodata.getNumberValue(1, "label0") -> 1337
Calling -> rarioData.setTextValue(1, "label0", "NEWVALUE", {"from": owner})
Transaction sent: 0x733cd0u35751d7a21d1b5b2acce6897fc09fcdf4897e3b9500f2c41371b
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 10
RarioData.setTextValue confirmed Block: 13492597 Gas used: 73559 (1.09%)
rariodata.getTextValue(1, "label0") -> 'NEWVALUE'
>>> rarioData.getAttribute("label1", {"from": user1})
File <console>, line 1, in <module>
File "brownie/network/contract.py", line 1728, in __call__
    return self.call(*args, block_identifier=block_identifier)
File "brownie/network/contract.py", line 1532, in call
    raise VirtualMachineError(e) from None
VirtualMachineError: revert: AccessControl: account 0x41d8d960d58ea1ebf4f8f7487fab323e177e76a3 is missing role 0x23a704056dc017bcf83bed8b68c59416dac1119pe77755efc3bde0a64e46e0c
>>> rarioData.getAttributeByIndex(1, {"from": user1})
File <console>, line 1, in <module>
File "brownie/network/contract.py", line 1728, in __call__
    return self.call(*args, block_identifier=block_identifier)
File "brownie/network/contract.py", line 1532, in call
    raise VirtualMachineError(e) from None
VirtualMachineError: revert: AccessControl: account 0x41d8d960d58ea1ebf4f8f7487fab323e177e76a3 is missing role 0x23a704056dc017bcf83bed8b68c59416dac1119pe77755efc3bde0a64e46e0c

```

4.6 RARIOMARKETPLACE CONTRACT

The contract `RarioMarketplace` inherits from `Marketplace`.

The contract contains the following functions:

- From `Marketplace`
 - `_cancelListing(uint256)` (private)
 - `_createListing(uint256,uint256,string,address,address,address,uint8)` (private)
 - `_distributeFunds(address,string,uint256)` (private)
 - `_finaliseListing(uint256)` (private)
 - `_getListingByListingId(uint256,bool)` (private)
 - `_isValidWallet(address)` (private)
 - `_reloadConfiguration()` (private)
 - `_sendValue(address,uint256,uint256)` (private)
 - `_transferToken(address,address,uint256)` (private)
 - `cancelListing(uint256)` (external)
 - `createAuction(uint256,uint256,string,address)` (external)
 - `createPrivateSale(uint256,uint256,string,address,address)` (external)
 - `createPublicSale(uint256,uint256,string,address)` (external)
 - `finalizeListing(uint256)` (external)
 - `getBidPriceInCurrency(uint256,string)` (external)
 - `getFinalisedListing(uint256)` (external)
 - `getListing(uint256)` (external)
 - `getListingBytokenId(uint256)` (external)
 - `getPendingWithdrawal(address)` (public)
 - `initializeMarketplace(address,address)` (internal)
 - `placeBid(uint256,address,string,uint256)` (external)
 - `reloadConfiguration()` (external)
 - `setMarketplaceConfig(address)` (external)
 - `updateListing(uint256,uint256)` (external)
 - `withdraw()` (public)
 - `withdrawFor(address)` (public)
- From `RarioMarketplace`

- `initialize(address,address,address)` (external)

Test 1: Creating a public sale

In this test, user1 created a public sale by calling the `createPublicSale` function. Initially, the price set was 1337. But then user1 called `updateListing` and updated the price to 3000 USD. Then, user2 bought the token successfully by calling the `placeBid` function. In the image below, we can see how the 95% of the tokens were transferred to user1 and the 5% to the treasury address:

```

token.ownerOf() -> 0x31798Bfcf3292015094bd033daB4le09C16c54ef
user1.address -> 0x31798Bfcf3292015094bd033daB4le09C16c54ef

Calling -> rarioUsers.addUser(1, user1.address, {'from': owner})
Transaction sent: 0x73042ab57842e623fa98d5f18c1a675146324dbd7a6c84d03cf521f1283656
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 27
RarioUsers.addUser confirmed Block: 13494214 Gas used: 132294 (1.97%)

Calling -> rarioUsers.addUser(2, user2.address, {'from': owner})
Transaction sent: 0x52c7b4f50a5f72d6e80e8e108be0955e3c629473c12713ec51401e10f42e
Gas price: 0.0 gwei Gas limit: 72d2915 Nonce: 28
RarioUsers.addUser confirmed Block: 13494215 Gas used: 117294 (1.74%)

Calling -> token.approve(rarioMarketplace.address, 1, {'from': user1})
Transaction sent: 0xe9154038dc528bee07a175cb44bb0c939057fc0d5a4123a1a12f093ea654137
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 0
RarioPolygonToken.approve confirmed Block: 13494216 Gas used: 48878 (0.73%)

token.ownerOf() -> 0x31798Bfcf3292015094bd033daB4le09C16c54ef
user1.address -> 0x31798Bfcf3292015094bd033daB4le09C16c54ef
rarioMarketplace.address -> 0xf93424a465597545fc0e032cd2d67FF536bc05B2
Calling -> rarioMarketplace.createPublicSale(1, 1337, user1.address, {'from': user1})
Transaction sent: 0x0094a57ec009424358c5e7ef4fb126879a377fc027f59aa6ff692565362d3c9
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
RarioMarketplace.createPublicSale confirmed Block: 13494217 Gas used: 325166 (4.84%)

token.ownerOf() -> 0x9342a465597545fc0e032cd2d67FF536bc05B2
user1.address -> 0x31798Bfcf3292015094bd033daB4le09C16c54ef
rarioMarketplace.address -> 0xf93424a465597545fc0e032cd2d67FF536bc05B2
rarioMarketplace.getListingByTokenId(1) -> (1, '0x31798Bfcf3292015094bd033daB4le09C16c54ef', '0x31798BfcF0000000000000000000000000000000000000000000000000000000000000000', 1337, 2592000, 900, 0)
rarioMarketplace.getListing(78541660797044910968829902406342334108369226379826116144642989268089806461) -> (1, 78541660797044910968829902406342334108369226379826116144642989268089806461, '0x31798BfcF0000000000000000000000000000000000000000000000000000000000000000', 1337, 2592000, 900, 0)
Calling -> rarioMarketplace.updateListing(78541660797044910968829902406342334108369226379826116144642989268089806461, 3000, {'from': user1})
Transaction sent: 0x17e990919e2ccf570cc14ac37f50b31c1d10582d2659154f338187fd
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 2
RarioMarketplace.updateListing confirmed Block: 13494218 Gas used: 35420 (0.53%)

rarioMarketplace.getListingByTokenId(1) -> (1, '0x31798Bfcf3292015094bd033daB4le09C16c54ef', '0x31798BfcF0000000000000000000000000000000000000000000000000000000000000000', 3000, 2592000, 900, 0)
rarioMarketplace.getListing(78541660797044910968829902406342334108369226379826116144642989268089806461) -> (1, 78541660797044910968829902406342334108369226379826116144642989268089806461, '0x31798BfcF0000000000000000000000000000000000000000000000000000000000000000', 3000, 2592000, 900, 0)

msg.value = ratioCurrencyManager.convert("USDT", "MATIC", 3000) -> 2000000000000000000000
$5% for user1 -> 1900000000000000000000
$5% for treasury -> 1000000000000000000000
token.ownerOf() -> 0x9342a465597545fc0e032cd2d67FF536bc05B2
rarioMarketplace.address -> 0x9342a465597545fc0e032cd2d67FF536bc05B2

Calling -> rarioMarketplace.placeBid(78541660797044910968829902406342334108369226379826116144642989268089806461, user2.address, "MATIC", 100, {'from': user2, 'value': msg.value})
Transaction sent: 0xb9c93cd0e691163d0837d0fde015fe0eeebd3793d5d11e7e9594e21be
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 0
RarioMarketplace.placeBid confirmed Block: 13494219 Gas used: 337588 (5.02%)

user1.balance() -> 118000000000000000000000000000000
treasury.balance() -> 101000000000000000000000000000000
user2.balance() -> 80000000000000000000000000000000
token.ownerOf() -> 0x0DE1B7DCf5edDCf18a37f7Bb6fa529B1644902B
user2.address -> 0x0DE1B7DCf5edDCf18a37f7Bb6fa529B1644902B

```

On the other hand, it is worth mentioning that the parameter `bidValue` is useless as long as the user calling `placeBid` is providing a `msg.value > 0`:

Listing 18: Marketplace.sol (Lines 308,310)

```

297 function placeBid(
298     uint256 listingId,
299     address buyerTokenWallet,
300     string calldata currency,
301     uint256 bidValue

```

```
Calling > rarioMarketplace.placeBid{txIndex:0x78514679870491968829902406323423798211616144644298926808906461, user2.address, "MATIC", 0, {'from': user2, 'value': msgValue})
    --> rarioMarketplace.placeBid{txIndex:0x78514679870491968829902406323423798211616144644298926808906461, user2.address, "MATIC", 0, {'from': user2, 'value': msgValue}}
Transaction sent: 0xa32316e963be93bbfc8d16c766a8220e542fc1ad6022a009e6b59172b4a695c
Gas price: 0.0 Gas limit: 671575 Nonce: 0
  RarioMarketplace.placeBid confirmed Block: 13494351 Gas used: 337576 (5.02%)
<transacton `0xa921f63fe3bcf81eb766a82f0e542fc1ad6022a009e6b59172b4a695c`>
--> output.read("user1.balance") -> "+ str(user1.balance())
output.read("treasury.balance") -> "+ str(treasury.balance())
output.read("user2.balance") -> "+ str(user2.balance())
output.read("token.ownerOf(1)") -> "+ str(token.ownerOf(1))
output.read("user2.address") -> "+ str(user2.address)
user1.balance() -> 119000000000000000000000000000000
treasury.balance() -> 10100000000000000000000000000000
user2.balance() -> 80000000000000000000000000000000
token.ownerOf(1) -> 0x7f95573da5457653abff5f8e11c109eb6aa0c29df
user2.address -> 0x7f95573da5457653abff5f8e11c109eb6aa0c29df
```

The payment can also be done with an ERC20 token, in this case, with USDC ERC20 tokens:

Test 2: Creating a private sale

In this test, user1 created a private sale by calling the `createPrivateSale` function setting the user2 as the buyer. The price set was 3000. Then, user2 bought the token successfully by calling the `placeBid` function:

```

Minting tokenid 1 to user1
Transaction sent: 0xe9c90e91371c2f110e0ee1b897a3f1d272ab9aaacb9f3f5a9e3352808119655e
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 26
RarioPolygonToken.mint confirmed Block: 13498056 Gas used: 134496 (2.00%)
token.ownerOf(1) -> 0x1d01835a7de31eacFd16FFbF2307263eAE095218
user1.address -> 0x1d01835a7de31eacFd16FFbF2307263eAE095218

Calling -> rariousers.addUser(1, user1.address, {'from': owner})
Transaction sent: 0xd77bd23ebbd1d751a1dd486660ca9e9739cad35f215c836b137fs24claca
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 27
RarioUsers.addUser confirmed Block: 13498057 Gas used: 132294 (1.97%)
RarioUsers.addUser confirmed Block: 13498058 Gas used: 117254 (1.74%)

Calling -> rariousers.addUser(2, user2.address, {'from': owner})
Transaction sent: 0xa027ec10e50e235a3de008c1fcd390bb0bba99ae195c192d05ac0d748a
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 28
RarioUsers.addUser confirmed Block: 13498058 Gas used: 117254 (1.74%)

Calling -> token.approve(rarioMarketplace.address, 1, {'from': user1})
Transaction sent: 0x6a056105d156082794dec1ed6cc59cb11a3e1a64e64096dase053ab578cf06f02
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 0
RarioPolygonToken.approve confirmed Block: 13498059 Gas used: 48878 (0.73%)
token.ownerOf(1) -> 0x1d01835a7de31eacFd16FFbF2307263eAE095218
user1.address -> 0x1d01835a7de31eacFd16FFbF2307263eAE095218
Calling -> rarioMarketplace.createPrivateSale("USD", user2.address, user2.address, {'from': user1})
Transaction sent: 0x7854166079704491096882990240634233108369022637982611616144642989268089806461, 1, '0x1d01835a7de31eacFd16FFbF2307263eAE095218', '0x1d01835a7de31eacFd16FFbF2307263eAE095218'
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
RarioMarketplace.createPrivateSale confirmed Block: 13498060 Gas used: 325581 (4.04%)
token.ownerOf(1) -> 0x0e98430e2d08fae4fc5989ffcc72leFl9d047bd
user1.address -> 0x1d01835a7de31eacFd16FFbF2307263eAE095218
rarioMarketplace.address -> 0x0f98430e2d08fae4fc5989ffcc72leFl9d047bd
RarioMarketplace.getListing("USD", "USDC", 3000, "USD", user2.address, user2.address, {'from': user1})
Transaction sent: 0x7854166079704491096882990240634233108369022637982611616144642989268089806461, 1, '0x1d01835a7de31eacFd16FFbF2307263eAE095218', '0x1d01835a7de31eacFd16FFbF2307263eAE095218', '0x0e98430e2d08fae4fc5989ffcc72leFl9d047bd', '0x0e98430e2d08fae4fc5989ffcc72leFl9d047bd', 3000, 2592000, 900, 0)
rarioMarketplace.getListings() -> 0x7854166079704491096882990240634233108369022637982611616144642989268089806461, 1, '0x1d01835a7de31eacFd16FFbF2307263eAE095218', '0x0e98430e2d08fae4fc5989ffcc72leFl9d047bd', '0x0e98430e2d08fae4fc5989ffcc72leFl9d047bd', 3000, 2592000, 900, 0)
USDC_AMOUNT = rarioCurrencyManager.convert("USD", "USDC", 3000) -> 29994001
Transaction sent: 0x8eadc5b3449fae57d939ec37e70201ce042cf04ad00ff2af2c5e7e50d5dc5b342f6
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 29
ERC20Token.transfer confirmed Block: 13498061 Gas used: 51061 (0.76%)
Transaction sent: 0x1042e7ad8237bec38ebfd11d66dbfb5bab94706da8ff7ff230197ef6d999
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 0
ERC20Token.approve confirmed Block: 13498062 Gas used: 44127 (0.66%)
usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(treasury.address) -> 0
usdc.balanceOf(user3.address) -> 29994002
Calling -> rarioMarketplace.placeBid("USD", "USDC", 3000, "USD", user2.address, {"from": user2})
Transaction sent: 0x7854166079704491096882990240634233108369022637982611616144642989268089806461, user2.address, "USDC", USDC_AMOUNT + 1, {'from': user2})
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
RarioMarketplace.placeBid confirmed Block: 13498063 Gas used: 342563 (5.10%)
usdc.balanceOf(user1.address) -> 20494302
usdc.balanceOf(treasury.address) -> 1499700
usdc.balanceOf(user2.address) -> 0
token.ownerOf(1) -> 0x0f03d5e00907d51cbe8656eA8767C0a6dfbb8202
user2.address -> 0x0f03d5e00907d51cbe8656eA8767C0a6dfbb8202

```

We can see how the sale is restricted just for the user2:

```

usdc.balanceOf(user1.address) -> 0
usdc.balanceOf(treasury.address) -> 0
usdc.balanceOf(user3.address) -> 29994002
Calling -> rarioMarketplace.placeBid("USD", "USDC", 3000, "USD", user2.address, {"from": user2})
Transaction sent: 0x827695dd278e2f2f1bcal32343dd2dc8a62de70737d5375ab3fe26f37629480
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
RarioMarketplace.placeBid confirmed (NFTMarketplace: Private Sale) Block: 134980197 Gas used: 93500 (1.39%)

```

Test 3: Creating an auction, payment with NATIVE currency

In this test, user1 created an auction sale by calling the `createAuction` function. Initially the price set was 3000. But then user1 called `updateListing` and updated the price to 5000 USD. Then multiple users placed a bid using `NATIVE` currency:

In the image above we can appreciate how the ETH amount was discounted from their balance and sent to the smart contract right after placing the highest bid. We can also see how the bid was returned also to the outbid users. We have also made sure that the seller cannot place any bid in his listing:

```

Calling -> rarioMarketplace.createAuction(1, 3000, "USDP", user1.address, {'from': user1})
Transaction sent: 0x2b1570591461552ee3e78e2335bd048d8e9e85f6c7be0646fc0005091d2681b
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
RarioMarketplace.createAuction confirmed Block: 13504346 Gas used: 325220 (1.84%)
token.ownerOf(1) -> 0x96c776120088314f568e4757fb41eb75f869b4e
user1.address -> 0xa0299fa741579bae9c525c2c574a055790c39e8ab
rarioMarketplace.getListing(78541660797044910968829902406342334108369226379826116161446442989268089006461, 1, '0xa0299fa741579bae9c525c2574a055790c39e8ab', '0xa0299fa741579bae9c525c2574a055790c39e8ab')
rarioMarketplace.getListing(78541660797044910968829902406342334108369226379826116161446442989268089006461, 2, '0xa0299fa741579bae9c525c2574a055790c39e8ab', '0xa0299fa741579bae9c525c2574a055790c39e8ab')
Calling -> rarioMarketplace.updateListing(listingID, 5000, {'from': user1})
Transaction sent: 0x94438c19c0bc26bbc8a4a54ff967b7b9884e851fb5a0ed50191491d4441ff
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 2
RarioMarketplace.updateListing confirmed Block: 13504347 Gas used: 35420 (0.53%)
rarioMarketplace.getListingByTokenId(1) -> (2, 78541660797044910968829902406342334108369226379826116161446442989268089006461, 1, '0xa0299fa741579bae9c525c2574a055790c39e8ab', '0xa0299fa741579bae9c525c2574a055790c39e8ab')
rarioMarketplace.getListing(78541660797044910968829902406342334108369226379826116161446442989268089006461, 2, '0xa0299fa741579bae9c525c2574a055790c39e8ab', '0xa0299fa741579bae9c525c2574a055790c39e8ab')
rarioMarketplace.getListing(78541660797044910968829902406342334108369226379826116161446442989268089006461, 3, '0xa0299fa741579bae9c525c2574a055790c39e8ab', '0xa0299fa741579bae9c525c2574a055790c39e8ab')
Calling -> rarioMarketplace.placeBid(78541660797044910968829902406342334108369226379826116161446442989268089006461, user1.address, "MATIC", 0, {'from': user1, 'value': MSGVALUE1 + 1})
Transaction sent: 0xdec02b97e0cc4d09d1470acb3e1003f674ea61cace5b694edc079c2d9e
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 3
RarioMarketplace.placeBid confirmed (RarioMarketplace: Cannot buy own listing) Block: 13504348 Gas used: 25555 (0.38%)

```

Test 4: Creating an auction, payment with TOKEN currency

In this test, user1 created an auction sale by calling the `createAuction` function. Initially, the price set was 3000. But then user1 called `updateListing` and updated the price to 5000 USD. Then multiple users placed a bid using `TOKEN` currency. 2 different issues were found which are explained in:

- INCORRECT LOGIC LEADS TO DOS IN AUCTION SALES
- OUTBID USER DOES NOT RECEIVE BACK HIS FUNDS

4.7 RARIOMARKETPLACECONFIG CONTRACT

The contract `RarioMarketplaceConfig` inherits from `MarketplaceConfig` and is used to configure different parameters used by the `RarioMarketplace` contract.

The contract `RarioMarketplaceConfig` contains the following functions:

- From `MarketplaceConfig`
 - `_existsUserWallet(address)` (internal)
 - `_setAllowExternalWallets(bool)` (private)
 - `_setAuctionExtensionDuration(uint256)` (private)
 - `_setAuctionPercentIncrementInBasisPoints(uint256)` (private)
 - `_setGasLimits(uint256,uint256,uint256)` (private)
 - `_setListingExpiry(uint256)` (private)
 - `_setPlatformFees(uint256)` (private)
 - `_setTokenContract(address)` (private)
 - `_setTreasury(address)` (private)
 - `_setUserDB(address)` (private)
 - `getAllowExternalWallets()` (external)
 - `getAuctionExtensionDuration()` (external)
 - `getAuctionPercentIncrementInBasisPoints()` (external)
 - `getGasLimits()` (external)
 - `getListingExpiry()` (external)
 - `getPlatformFees()` (external)
 - `getTokenAddress()` (external)
 - `getTreasuryAddress()` (external)
 - `getUserDBAddress()` (external)
 - `initializeMarketplaceConfig(address,address,address)` (internal)
 - `isValidWallet(address)` (external)
 - `setAllowExternalWallets(bool)` (external)
 - `setAuctionExtensionDuration(uint256)` (external)
 - `setAuctionPercentIncrementInBasisPoints(uint256)` (external)
 - `setGasLimits(uint256,uint256,uint256)` (external)
 - `setListingExpiry(uint256)` (external)

- `setPlatformFees(uint256)` (external)
- `setTokenAddress(address)` (external)
- `setTreasuryAddress(address)` (external)
- `setUserDBAddress(address)` (external)
- From `RarioMarketplaceConfig`
 - `initialize(address,address,address,address)` (external)

Test: `setAllowExternalWallets` - True, `setPlatformFees` - 5000 (50%)

In this test, we have set the following configuration in the `MarketplaceConfig` contract:

- `marketplaceConfig.setAllowExternalWallets(True, {'from': owner})`
- `marketplaceConfig.setPlatformFees(5000, {'from': owner})`

Below we can see, how the platform fee is now 50% as the 50% of the payment goes to the treasury address. On the other hand, we can also see how external wallets are allowed to bid without having added them previously in the `RarioUsers` contract:

```

Calling -> marketplaceConfig.setAllowExternalWallets(True, {'from': owner})
Transaction sent: 0x1f1d0e5f3c03716b33b36733801296515a7c19a1d4d715a9561e36e21f45ac45
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 27
RarioMarketplace.setAllowExternalWallets confirmed Block: 13504733 Gas used: 31845 (0.47%)

Calling -> marketplaceConfig.setPlatformFees(5000, {'from': owner})
Transaction sent: 0x82d625691e7a9ccce3233ca1fc6b399fdad100201e15275ecb262c9148ecb
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 28
RarioMarketplace.setPlatformFees confirmed Block: 13504734 Gas used: 31875 (0.47%)

Calling -> rarioMarketplace.reloadConfiguration({'from': owner})
Transaction sent: 0x3be1eb0b35b7342f2fe0043b1e1c175336e99c756e0502e2074d13a178e49lc
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 29
RarioMarketplace.reloadConfiguration confirmed Block: 13504735 Gas used: 157029 (2.34%)

Calling -> rarioMarketplace.createPublicSale(1, 1337, "USD", user1.address, {'from': user1})
Transaction sent: 0x34625c88532d210bcb5a5ab70564e54eeefcc5110e23b351779ed7bc9c94dc661
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
RarioMarketplace.createPublicSale confirmed Block: 13504736 Gas used: 318241 (4.73%)

msg.value = rarioCurrencyManager.convert("USD", "MATIC", 3000) -> 200000000000000000000000
for user1 = 100000000000000000000000000000000
for treasury = 100000000000000000000000000000000
token.ownerOf(1) -> 0x1798Bb594b28963FA73d2b40e63506a02Ee79E9
rarioMarketplace.address -> 0x1798Bb594b28963FA73d2b40e63506a02Ee79E9

Calling -> rarioMarketplace.placeBid(785416670704431096882990240634233410836926379826116161446442989268089806461, user2.address, "MATIC", 0, {'from': user2, 'value': msg.value})
Transaction sent: 0x17d7d45b15b5076577836547ba110665f5d1064d9fa8fa1fc54f24b01541c5
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 0
RarioMarketplace.placeBid confirmed Block: 13504737 Gas used: 330651 (4.92%)

user1.balance() -> 11000000000000000000000000000000
treasury.balance() -> 11000000000000000000000000000000
user2.balance() -> 80000000000000000000000000000000
rarioMarketplace.getSale(1) -> 0x4d7df486c8c22965d4953BbDA011b2c1B54B05
token.ownerOf(1) -> 0x4d7df486c8c22965d4953BbDA011b2c1B54B05
user2.address -> 0x4d7df486c8c32c29968d4953BbDA011b2c1B54B05

```

4.8 RARIOPOLYGONTOKEN CONTRACT

The contract `RarioPolygonToken` inherits from `RarioPolygonNFT`. `RarioPolygonNFT` at the same time inherits from `RarioNFT` contract.

The contract `RarioPolygonToken` contains the following functions:

- From `RarioPolygonNFT`
 - `deposit(address,bytes)` (external)
 - `encodeTokenMetadata(uint256)` (external)
 - `initializeRarioPolygonNFT(string,string,address,address)` (internal)
 - `withdraw(uint256)` (external)
 - `withdrawBatch(uint256[])` (external)
 - `withdrawWithMetadata(uint256)` (external)
- From `RarioNFT`
 - `existsAttribute(string)` (external)
 - `getAttribute(string)` (external)
 - `getAttributeBooleanValue(uint256,string)` (external)
 - `getAttributeByIndex(uint256)` (external)
 - `getAttributeCount()` (external)
 - `getAttributeNumberValue(uint256,string)` (external)
 - `getAttributeTextValue(uint256,string)` (external)
 - `initializeRarioNFT(string,string,address)` (internal)
 - `isApprovedForAll(address,address)` (public)
 - `isContractApprovedForAll(address)` (external)
 - `mint(address,uint256)` (external)
 - `safeMint(address,uint256)` (external)
 - `safeMint(address,uint256,bytes)` (external)
 - `setContractApprovalForAll(address,bool)` (external)
- From `RarioPolygonToken`
 - `initialize(address,address,address,string,string)` (external)

The contract `RarioPolygonNFT` correctly follows the [Polygon Mintable Assets documentation](#):

1. The contract `RarioPolygonNFT` has the same structure as `ChildMintableERC721.sol`
2. The contract `RarioPolygonNFT` contains the functions `deposit`, `withdraw` and `mint`. The `mint` function is inherited, though, from `RarioNFT` contract and can only be called by an OPERATOR.

Listing 19: RarioNFT.sol (Lines 40)

```
39     function mint(address to, uint256 tokenId) external virtual {  
40         _checkRole(OPERATOR, msgSender());  
41         _mint(to, tokenId);  
42     }
```

It is worth mentioning, as described in the [MINT FUNCTION IN RARIOPOLYGNFT DOES NOT VERIFY IF TOKEN WAS WITHDRAWN](#) findings page that the function `mint` in `RarioPolygonNFT` does not verify if the token was withdrawn previously as it is done in `ChildMintableERC721.sol`

4.9 RARIOETHEREUMTOKEN CONTRACT

The contract `RarioEthereumToken` inherits from `RarioEthereumNFT`. `RarioEthereumNFT` at the same time inherits from `RarioNFT` contract.

The contract `RarioEthereumToken` contains the following functions:

- From `RarioEthereumNFT`
 - `exists(uint256)` (external)
 - `initializeRarioEthereumNFT(string, string, address, address)` (internal)
 - `mint(address, uint256)` (external)
 - `mint(address, uint256, bytes)` (external)
 - `setTokenMetadata(uint256, bytes)` (internal)
 - `version()` (external)
- From `RarioNFT`
 - `existsAttribute(string)` (external)
 - `getAttribute(string)` (external)
 - `getAttributeBooleanValue(uint256, string)` (external)
 - `getAttributeByIndex(uint256)` (external)
 - `getAttributeCount()` (external)
 - `getAttributeNumberValue(uint256, string)` (external)
 - `getAttributeTextValue(uint256, string)` (external)
 - `initializeRarioNFT(string, string, address)` (internal)
 - `isApprovedForAll(address, address)` (public)
 - `isContractApprovedForAll(address)` (external)
 - `safeMint(address, uint256)` (external)
 - `safeMint(address, uint256, bytes)` (external)
 - `setContractApprovalForAll(address, bool)` (external)
- From `RarioEthereumToken`
 - `initialize(address, address, address, string, string)` (external)

The contract `RarioEthereumNFT` correctly follows the [Polygon Mintable Assets documentation](#):

1. The contract `RarioEthereumNFT` has the same structure as `DummyMintableERC721.sol`

4.10 RARIOUSERS CONTRACT

The contract `RarioUsers` inherits from `UserDB`. This contract is used like a database to keep control of all the users and wallets.

The contract `RarioUsers` contains the following functions which are all only callable by the OPERATOR role:

- From UserDB
 - `addUser(uint256,address)` (external)
 - `existsUserId(uint256)` (external)
 - `existsUserWallet(address)` (external)
 - `getUserByAddress(address)` (external)
 - `getUserById(uint256)` (external)
 - `getUserByIndex(uint256)` (external)
 - `getUserCount()` (external)
 - `updateUser(uint256,address)` (external)
- From RarioUsers
 - `initialize(address)` (external)

```
>>> rariousers.addUser(1, user1.address, {'from': owner})
Transaction sent: 0x7773e83162c90a8cd0304a45ff88b185975d8b693c92134e1e6e18a9c5f1af9
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 21
RarioUsers.addUser confirmed Block: 13511177 Gas used: 132294 (1.97%)
<Transaction '0x7773e83162c90a8cd0304a45ff88b185975d8b693c92134e1e6e18a9c5f1af9'>
>>> rariousers.addUser(2, user2.address, {'from': owner})
Transaction sent: 0xb67081182905b407737c2bd8f98924e8adbc47160d22f7fb38cf50af02079e
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 28
RarioUsers.addUser confirmed Block: 13511178 Gas used: 117294 (1.74%)
<Transaction '0xb67081182905b407737c2bd8f98924e8adbc47160d22f7fb38cf50af02079e'>
>>> rariousers.existsUserId(0)
False
>>> rariousers.existsUserId(1)
True
>>> rariousers.existsUserId(2)
True
>>> rariousers.existsUserWallet(user1.address)
True
>>> rariousers.existsUserWallet(user2.address)
True
>>> rariousers.existsUserWallet(user3.address)
False
>>> rariousers.getUserCount()
1
>>> rariousers.getUserByIndex(1)
(1, "0x3BDE9BzP0f540e3Bd46cc679bb288E01D0D61B")
>>> rariousers.getUserByIndex(0)
(0, "0x1B1c0e70CBFcId1b125c67a4#bF010b7E5196C3")
>>> rariousers.getUserById(1)
(1, "0x1B1c0e70CBFcId1b125c67a4#bF010b7E5196C3")
>>> rariousers.getUserById(0)
(0, "0x3BDE9BzP0f540e3Bd46cc679bb288E01D0D61B")
>>> rariousers.updateUser(2, user1.address)
Transaction sent: 0xb1e17a4a9c1e2848669a4a5f0b1abf3ca1d1bb4c45bb1b5d13df1216b5053f
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 29
RarioUsers.updateUser confirmed Block: 13511179 Gas used: 29009 (0.43%)
<Transaction '0xb1e17a4a9c1e2848669a4a5f0b1abf3ca1d1bb4c45bb1b5d13df1216b5053f'>
>>> rariousers.getUserById(1)
(1, "0x1B1c0e70CBFcId1b125c67a4#bF010b7E5196C3")
>>> rariousers.getUserById(2)
(2, "0x1B1c0e70CBFcId1b125c67a4#bF010b7E5196C3")
>>> rariousers.getUserByAddress(user1.address)
(1, "0x1B1c0e70CBFcId1b125c67a4#bF010b7E5196C3")
```

All the functions are working as expected as can be seen above except `updateUser` which is not validating if the address of the wallet was

already added, as described in the findings page UPDATEUSER FUNCTION MISSING REQUIRE STATEMENT.

AUTOMATED TESTING

5.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the scoped contracts. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified all the contracts in the repository and was able to compile them correctly into their abi and binary formats, Slither was run on the all-scoped contracts. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.

Slither results:

RarioCurrencyManager.sol

```
ERC1967Upgrade._functionDelegateCall(address,bytes) (contracts/traitz/ERC1967Upgrade.sol#200-206) uses delegatecall to a input-controlled function id
  - (success, returnData) = target.delegatecall(data) (contracts/traitz/ERC1967Upgrade.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

RarioCurrencyManager (contracts/RarioCurrencyManager.sol#13-24) is an upgradable contract that does not protect its initialize functions: RarioCurrencyManager.initialize(address,string) (contracts/RarioCurrencyManager.sol#20-23). Anyone can call it by sending a transaction to the address of the contract and specifying the function selector. Documentation#unprotected-upgradeable-contract

CurrencyManager._convert(string,uint256) price (contracts/traitz/CurrencyManager.sol#347) is a local variable never initialized
CurrencyManager._Convert(string,uint256,uint256) currencyFairDecimals (contracts/traitz/CurrencyManager.sol#345) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

AccessControl._grantRole(bytes32,address) (contracts/traitz/AccessControl.sol#227-233) ignores return value by _roleMembers[role].add(account) (contracts/traitz/AccessControl.sol#230)
AccessControl._revokeRole(bytes32,address) (contracts/traitz/AccessControl.sol#235-241) ignores return value by _roleMembers[role].remove(account) (contracts/traitz/AccessControl.sol#238)
CurrencyManager.addCurrencyType(string,uint256) (contracts/traitz/CurrencyManager.sol#325-330) ignores return value by _feeFeeFeededContractAddress (contracts/traitz/CurrencyManager.sol#344-345)
CurrencyManager._Convert(string,uint256) (contracts/traitz/CurrencyManager.sol#328-344) ignores return value by _IPriceFeeFeededContractAddress (contracts/traitz/CurrencyManager.sol#344-345)
CurrencyManager._Convert(string,uint256,uint256) (contracts/traitz/CurrencyManager.sol#328-344) ignores return value by _IPriceFeeFeededContractAddress (contracts/traitz/CurrencyManager.sol#345-350)
CurrencyManager._addCurrencyType(string,uint256) (contracts/traitz/CurrencyManager.sol#328-344) ignores return value by _symbols.add(symbol) (contracts/traitz/CurrencyManager.sol#345-350)
CurrencyManager._addCurrencyType(string,uint256,uint256) (contracts/traitz/CurrencyManager.sol#328-344) ignores return value by _symbols.add(symbol) (contracts/traitz/CurrencyManager.sol#345-350)
CurrencyManager._removeCurrencyFair(string) (contracts/traitz/CurrencyManager.sol#397-410) ignores return value by _currencyFairSymbols.remove(symbol) (contracts/traitz/CurrencyManager.sol#404)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Variable `CurrencyManager._convert(string,uint256)` currencyFairDecimals (contracts/traitz/CurrencyManager.sol#347) in CurrencyManager._convert(string,uint256) (contracts/traitz/CurrencyManager.sol#328-344) potentially used before declaration: ( $(amount * (toCurrencyDecimals + currencyFairDecimals - fromCurrencyDecimals)) / uint256(price))$  (contracts/traitz/CurrencyManager.sol#352-357)
Variable `CurrencyManager._Convert(string,uint256,uint256)` currencyFairDecimals (contracts/traitz/CurrencyManager.sol#328-344) in CurrencyManager._Convert(string,uint256) (contracts/traitz/CurrencyManager.sol#328-344) potentially used before declaration: ( $(amount * (toCurrencyDecimals + currencyFairDecimals - fromCurrencyDecimals)) / uint256(price)$ ) (contracts/traitz/CurrencyManager.sol#352-357)
Variable `CurrencyManager._Convert(string,uint256)` currencyFairDecimals (contracts/traitz/CurrencyManager.sol#328-344) in CurrencyManager._Convert(string,uint256) (contracts/traitz/CurrencyManager.sol#328-344) potentially used before declaration: ( $(amount * (toCurrencyDecimals + currencyFairDecimals - fromCurrencyDecimals)) / uint256(price)$ ) (contracts/traitz/CurrencyManager.sol#352-357)
Variable `CurrencyManager._addCurrencyType(string,uint256)` currencyFairDecimals (contracts/traitz/CurrencyManager.sol#328-344) in CurrencyManager._addCurrencyType(string,uint256) (contracts/traitz/CurrencyManager.sol#352-357)
Variable `CurrencyManager._removeCurrencyFair(string)` currencyFairDecimals (contracts/traitz/CurrencyManager.sol#397-410) in CurrencyManager._removeCurrencyFair(string) (contracts/traitz/CurrencyManager.sol#404) potentially used before declaration: ( $(amount * (toCurrencyDecimals + currencyFairDecimals - fromCurrencyDecimals)) / uint256(price)$ ) (contracts/traitz/CurrencyManager.sol#352-357)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-use-of-local-variables

Reentrancy in ERC1967Upgrade._upgradeToAndCall(address,bytes,bool) (contracts/traitz/ERC1967Upgrade.sol#82-107):
  - _functionDelegateCall(newImplementation,data) (contracts/traitz/ERC1967Upgrade.sol#82)
    - (success, returnData) = target.delegatecall(data) (contracts/traitz/ERC1967Upgrade.sol#204)
    - _functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address,oldImplementation)")) (contracts/traitz/ERC1967Upgrade.sol#100)
      - target.delegatecall(abi.encodeWithSignature("upgradeTo(address,oldImplementation)")) (contracts/traitz/ERC1967Upgrade.sol#204)
    Event emitted after the call(s):
      - _Upgraded(newImplementation) (contracts/traitz/ERC1967Upgrade.sol#105)
      - _Upgraded(newImplementation) (contracts/traitz/ERC1967Upgrade.sol#105)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
```

RarioData.sol

```

Address.isContract(address) (contracts/lib/Address.sol#26-36) uses assembly
  - INLINE ASM (contracts/lib/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#169-188) uses assembly
  - INLINE ASM (contracts/lib/Address.sol#170-187)
EnumerableSet.values(EnumerableSet.AddressesSet) (contracts/lib/EnumerableSet.sol#273-282) uses assembly
    - INLINE ASM (contracts/lib/EnumerableSet.sol#277-279)
EnumerableSet.contains(EnumerableSet.AddressesSet,bytes) (contracts/lib/EnumerableSet.sol#346-355) uses assembly
    - INLINE ASM (contracts/lib/EnumerableSet.sol#350-352)
StorageSlot.getAddressesSlot(bytes2) (contracts/lib/StorageSlot.sol#81-95) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#84-86)
StorageSlot.getBooleanSlot(bytes2) (contracts/lib/StorageSlot.sol#80-94) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#84-86)
StorageSlot.getUsize256Slot(bytes2) (contracts/lib/StorageSlot.sol#149-173) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#170-172)
StorageSlot.getUsize256Slot(bytes2) (contracts/lib/StorageSlot.sol#178-182) uses assembly
  - INLINE ASM (contracts/traitz/Context.sol#16-15)
Context.msgSender() (contracts/traitz/Context.sol#19-21)
  - INLINE ASM (contracts/traitz/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usec

Address.functionCall(addresses,bytes) (contract/lib/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(addresses,bytes,uint256) (contract/lib/Address.sol#105-114) is never used and should be removed
Address.functionCallWithValue(addresses,bytes,uint256,string) (contract/lib/Address.sol#122-131) is never used and should be removed
Address.functionCallStatic(addresses,bytes,uint256) (contract/lib/Address.sol#141-149) is never used and should be removed
Address.functionCallStatic(addresses,bytes,uint256,string) (contract/lib/Address.sol#158-167) is never used and should be removed
Address.setValue(address,uint256) (contract/lib/Address.sol#184-195) is never used and should be removed
Context.msgSender() (contracts/traitz/Context.sol#16-15) is never used and should be removed
  - INLINE ASM (contracts/traitz/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usec

Address.functionCallAddresses(bytes) (contract/lib/Address.sol#79-81) is never used and should be removed
Address.functionCallAddressesWithValue(addresses,bytes,uint256) (contract/lib/Address.sol#105-114) is never used and should be removed
Address.functionCallAddressesWithValue(addresses,bytes,uint256,string) (contract/lib/Address.sol#122-131) is never used and should be removed
Address.functionCallAddressesStatic(addresses,bytes,uint256) (contract/lib/Address.sol#141-149) is never used and should be removed
Address.functionCallAddressesStatic(addresses,bytes,uint256,string) (contract/lib/Address.sol#158-167) is never used and should be removed
Address.setValue(address,uint256) (contract/lib/Address.sol#184-195) is never used and should be removed
Context.msgSender() (contracts/traitz/Context.sol#16-15) is never used and should be removed
  - INLINE ASM (contracts/traitz/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usec

ERC1967Upgrades.authorizeUpgrade(address) (contracts/traitz/ERC1967Upgrade.sol#60) is never used and should be removed
ERC1967Upgrades.changeAdmin(address) (contracts/traitz/ERC1967Upgrade.sol#141-144) is never used and should be removed
ERC1967Upgrades.getAdmin() (contracts/traitz/ERC1967Upgrade.sol#141-144) is never used and should be removed
ERC1967Upgrades.setAdmin(address) (contracts/traitz/ERC1967Upgrade.sol#141-144) is never used and should be removed
ERC1967Upgrades.upgradeAndCall(addresses,bytes,bytes,bytes) (contracts/traitz/ERC1967Upgrade.sol#152-159) is never used and should be removed
ERC1967Upgrades.upgradeAndCall(address,bytes,bytes,bytes) (contracts/traitz/ERC1967Upgrade.sol#166-173) is never used and should be removed
EnumerableSet.add(EnumerableSet.UintSet,bytes2) (contracts/lib/EnumerableSet.sol#157-159) is never used and should be removed
EnumerableSet.add(EnumerableSet.UintSet,bytes2) (contracts/lib/EnumerableSet.sol#172-174) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,bytes) (contracts/lib/EnumerableSet.sol#204-206) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,bytes) (contracts/lib/EnumerableSet.sol#213-215) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,bytes) (contracts/lib/EnumerableSet.sol#218-220) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (contract/lib/EnumerableSet.sol#181-183) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (contract/lib/EnumerableSet.sol#196-198) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet,bytes2) (contracts/lib/EnumerableSet.sol#206-208) is never used and should be removed
EnumerableSet.setValue(EnumerableSet.AddressesSet,bytes) (contract/lib/EnumerableSet.sol#273-282) is never used and should be removed
StorageSlot.getByte32Slot(bytes2) (contracts/lib/StorageSlot.sol#149-173) is never used and should be removed
StorageSlot.getByte32Slot(bytes2) (contracts/lib/StorageSlot.sol#178-182) is never used and should be removed
Strings.equals(string,string) (contracts/lib/Strings.sol#67-71) is never used and should be removed
Strings.toHexString(uint256) (contracts/lib/Strings.sol#199-205) is never used and should be removed
Strings.toHexString(uint256) (contracts/lib/Strings.sol#199-205) is never used and should be removed
  - INLINE ASM (contracts/lib/Strings.sol#199-205)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usec

ERC1967Upgrades.functionDelegateCall(address,bytes) (contracts/traitz/ERC1967Upgrade.sol#200-206) uses delegatecall to a input-controlled function id
  - (success,returndata) = target.delegatecall(data) (contracts/traitz/ERC1967Upgrade.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

RarioData (contracts/RarioData.sol#13-23) is an upgradable contract that does not protect its initialize functions: RarioData.initialize(address) (contracts/RarioData.sol#20-23). Anyone can delete the contract with: ERC1962UUUPS.upgradeTo(address) (contracts/RarioData.sol#13-23)ERC1962UUUPS.upgradeTo(address,bytes) (contracts/traitz/ERC1967Upgrade.sol#45-46)Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unprotected-upgradeable-contract
  - 
AccessControl.grantRole(bytes32,address) (contracts/traitz/AccessControl.sol#47-53) ignores return value by _setRoleMembers(); add(account) (contracts/traitz/AccessControl.sol#230)
AccessControl.setRoleAdmin(bytes32,address) (contracts/traitz/AccessControl.sol#173-179) ignores return value by _setRoleAdmin(); add(account) (contracts/traitz/AccessControl.sol#230)
MetadataAB.addAttribute(string,string,MetadataAB.AttributeType,bool) (contracts/traitz/MetadataAB.sol#172-189) ignores return value by _attributeNames.add(attribute) (contracts/traitz/MetadataAB.sol#191)
MetadataAB.setTextValue(uint256,string,string) (contracts/traitz/MetadataAB.sol#204-221) ignores return value by _tokenIds.add(tokenId) (contracts/traitz/MetadataAB.sol#217)
MetadataAB.setBooleanValue(uint256,bool) (contracts/traitz/MetadataAB.sol#226-235) ignores return value by _tokenIds.add(tokenId) (contracts/traitz/MetadataAB.sol#255)
MetadataAB.setBooleanValue(uint256,string,bool) (contracts/traitz/MetadataAB.sol#242-259) ignores return value by _tokenIds.add(tokenId) (contracts/traitz/MetadataAB.sol#255)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usec

Reentrancy in ERC1967Upgrades.upgradeToAndCall(id,source,address,bytes,bool) (contracts/traitz/ERC1967Upgrade.sol#82-107):
  - _functionImplementationCall(implementation,data) (contracts/traitz/ERC1967Upgrade.sol#82)
    - (success,returndata) = target.delegatecall(data) (contracts/traitz/ERC1967Upgrade.sol#82)
      - _functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address,bytes)",source)) (contracts/traitz/ERC1967Upgrade.sol#82)
        - (success,returndata) = target.delegatecall(data) (contracts/traitz/ERC1967Upgrade.sol#82)
      Event emitted(_id,source,callType)
        - _UpgradedImplementation (contracts/traitz/ERC1967Upgrade.sol#82)
          - _upgradeTo(newImplementation) (contracts/traitz/ERC1967Upgrade.sol#100)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/lib/Address.sol#26-36) uses assembly
  - INLINE ASM (contracts/lib/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#169-188) uses assembly
  - INLINE ASM (contracts/lib/Address.sol#170-187)
EnumerableSet.values(EnumerableSet.AddressesSet) (contracts/lib/EnumerableSet.sol#273-282) uses assembly
  - INLINE ASM (contracts/lib/EnumerableSet.sol#277-279)
EnumerableSet.contains(EnumerableSet.AddressesSet,bytes) (contracts/lib/EnumerableSet.sol#346-355) uses assembly
  - INLINE ASM (contracts/lib/EnumerableSet.sol#350-352)
StorageSlot.getAddressSlot(bytes2) (contracts/lib/StorageSlot.sol#81-95) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#84-86)
StorageSlot.getBooleanSlot(bytes2) (contracts/lib/StorageSlot.sol#80-94) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#84-86)
StorageSlot.getUsize256Slot(bytes2) (contracts/lib/StorageSlot.sol#149-173) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#170-172)
StorageSlot.getUsize256Slot(bytes2) (contracts/lib/StorageSlot.sol#178-182) uses assembly
  - INLINE ASM (contracts/traitz/Context.sol#16-15)
Context.msgSender() (contracts/traitz/Context.sol#19-21)
  - INLINE ASM (contracts/traitz/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usec

Address.functionCall(addresses,bytes) (contracts/lib/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(addresses,bytes,uint256) (contracts/lib/Address.sol#105-114) is never used and should be removed
Address.functionCallWithValue(addresses,bytes,uint256,string) (contracts/lib/Address.sol#122-131) is never used and should be removed
Address.functionCallStatic(addresses,bytes,uint256) (contracts/lib/Address.sol#141-149) is never used and should be removed
Address.functionCallStatic(addresses,bytes,uint256,string) (contracts/lib/Address.sol#158-167) is never used and should be removed
Address.setValue(address,uint256) (contracts/lib/Address.sol#184-195) is never used and should be removed
Context.msgSender() (contracts/traitz/Context.sol#16-15) is never used and should be removed
  - INLINE ASM (contracts/traitz/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usec

ERC1965Interface.initialize(ERC1965Interface) (contracts/traitz/ERC1965Interface.sol#46-52) is never used and should be removed
ERC1962UUUPS.authorizeUpgrade(address) (contracts/traitz/ERC1962UUUPS.sol#60) is never used and should be removed
ERC1962UUUPS.setRoleAdmin(bytes32,address) (contracts/traitz/ERC1962UUUPS.sol#173-179) ignores return value by _setRoleAdmin(); add(account) (contracts/traitz/AccessControl.sol#230)
ERC1967Upgrades.getAdmin() (contracts/traitz/ERC1967Upgrade.sol#141-144) is never used and should be removed
ERC1967Upgrades.setAdmin(address) (contracts/traitz/ERC1967Upgrade.sol#141-144) is never used and should be removed
ERC1967Upgrades.upgradeAndCall(addresses,bytes,bytes,bytes) (contracts/traitz/ERC1967Upgrade.sol#152-159) is never used and should be removed
ERC1967Upgrades.upgradeAndCall(address,bytes,bytes,bytes) (contracts/traitz/ERC1967Upgrade.sol#166-173) is never used and should be removed
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes2) (contracts/lib/EnumerableSet.sol#157-159) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes) (contracts/lib/EnumerableSet.sol#204-206) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes) (contracts/lib/EnumerableSet.sol#213-215) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes) (contracts/lib/EnumerableSet.sol#218-220) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (contract/lib/EnumerableSet.sol#320-322) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes) (contracts/lib/EnumerableSet.sol#167-169) is never used and should be removed
EnumerableSet.setValue(EnumerableSet.Bytes32Set,bytes) (contracts/lib/EnumerableSet.sol#273-282) is never used and should be removed
EnumerableSet.setValue(EnumerableSet.Bytes32Set,bytes) (contracts/lib/EnumerableSet.sol#277-279)
EnumerableSet.setValue(EnumerableSet.Bytes32Set,bytes) (contracts/lib/EnumerableSet.sol#207-209) is never used and should be removed
EnumerableSet.setValue(EnumerableSet.Bytes32Set,bytes) (contracts/lib/EnumerableSet.sol#218-220) is never used and should be removed
EnumerableSet.setString(EnumerableSet.StringSet,bytes) (contracts/lib/EnumerableSet.sol#127-129) is never used and should be removed
EnumerableSet.setString(EnumerableSet.StringSet,bytes) (contracts/lib/EnumerableSet.sol#127-129)
StorageSlot.getUsize256Slot(bytes2) (contracts/lib/StorageSlot.sol#170-172) is never used and should be removed
StorageSlot.getUsize256Slot(bytes2) (contracts/lib/StorageSlot.sol#178-182) is never used and should be removed
Strings.equals(string,string) (contracts/lib/Strings.sol#67-71) is never used and should be removed
Strings.toHexString(uint256) (contracts/lib/Strings.sol#199-205) is never used and should be removed
Strings.toHexString(uint256) (contracts/lib/Strings.sol#199-205) is never used and should be removed
  - INLINE ASM (contracts/lib/Strings.sol#199-205)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

```

```

RarioEthereumToken.sol
ERC1967Upgrade.functionDelegateCall(address,bytes) (contracts/trait/ERC1967Upgrade.sol#200-206) uses delegatecall to a input-controlled function id
  - (success,returnData) = target.delegatecall(data) (contracts/trait/ERC1967Upgrade.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

RarioEthereumToken (contracts/ERC1967Upgrade.sol#3-15) is an upgradeable contract that does not protect its initialize functions: RarioEthereumToken.initialize(address,address,address,string) (contracts/RarioEthereumToken.sol#14-16). Anyone can send the contract with ERC1967Upgrade.upgradeTo(address) (contracts/trait/ERC1967Upgrade.sol#3-9)(ERC1822UPRS.upgradeToAndCall(address,bytes)) (contracts/trait/ERC1822UPRS.sol#45-48) Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-upgrade

AccessControl.grantRole(bytes32,address) (contracts/trait/AccessControl.sol#27-23) ignores return value by roleMembers[role].add(account) (contracts/trait/AccessControl.sol#139)
AccessControl.revokeRole(bytes32,address) (contracts/trait/AccessControl.sol#123-24) ignores return value by roleMembers[role].remove(account) (contracts/trait/AccessControl.sol#138)
ERC721INFT._checkOnERC721Received(address,address,uint256,bytes) (contracts/trait/ERC721INFT.sol#438-459) ignores return value by IERC721Receiver(to).onERC721Received(msgSender(),from,to,tokenId_,data) (contracts/trait/ERC721INFT.sol#440-455)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

Variable ERC721INFT._checkOnERC721Received(address,address,uint256,bytes) (contracts/trait/ERC721INFT.sol#438-459) potential y used before declaration, return == IERC721Receiver.onERC721Received.selector (contracts/trait/ERC721INFT.sol#446)
Variable ERC721INFT._checkOnERC721Received(address,address,uint256,bytes) (contracts/trait/ERC721INFT.sol#438-459) potential y used before declaration, reason.length == 0 (contracts/trait/ERC721INFT.sol#449)
Variable ERC721INFT._checkOnERC721Received(address,address,uint256,bytes), reason (contracts/trait/ERC721INFT.sol#447) in ERC721INFT._checkOnERC721Received(address,address,uint256,bytes) (contracts/trait/ERC721INFT.sol#449) potential y used before declaration, revert(uint256,uint256) (2 + reason.length) (contracts/trait/ERC721INFT.sol#449)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-use-of-local-variables

Reentrancy in ERC1967Upgrade._upgradeToAndCall(call.value,address,bytes,boo) (contracts/trait/ERC1967Upgrade.sol#107):
  External calls:
    - functionDelegateCall(neuImplementation,data) (contracts/trait/ERC1967Upgrade.sol#92)
      - (success,returnData) = target.delegatecall(data) (contracts/trait/ERC1967Upgrade.sol#92)
        - functionUpgradeTo(neuImplementation) (contracts/trait/ERC1967Upgrade.sol#100)
          - (success,returnData) = target.delegatecall(data) (contracts/trait/ERC1967Upgrade.sol#100)
    Event emitted after the call(s):
      - _upgradeTo(neuImplementation) (contracts/trait/ERC1967Upgrade.sol#105)
        - _upgrade(neuImplementation) (contracts/trait/ERC1967Upgrade.sol#105)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/lib/Address.sol#26-36) uses assembly
  - INLINE ASM (contracts/lib/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#16-18) uses assembly
EnumerableSet.values(IEnumerableSet) (contracts/lib/EnumerableSet.sol#27-29)
  - INLINE ASM (contracts/lib/EnumerableSet.sol#27-29)
EnumerableSet.setValues(IEnumerableSet) (contracts/lib/EnumerableSet.sol#34-35) uses assembly
  - INLINE ASM (contracts/lib/EnumerableSet.sol#35)
StorageSlot.getAddressSlot(bytes32) (contracts/lib/StorageSlot.sol#15-15) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#15)
StorageSlot.getAddressSlot(bytes32) (contracts/lib/StorageSlot.sol#46-46) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#46)
StorageSlot.getBoolStorageSlot(bytes32) (contracts/lib/StorageSlot.sol#65-73) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#73)
StorageSlot.getUint256StorageSlot(bytes32) (contracts/lib/StorageSlot.sol#78-82) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#82)
Context._msgValue(address,bytes) (contracts/trait/Context.sol#19-21)
  - INLINE ASM (contracts/trait/Context.sol#19-21)
ERC721INFT._checkOnERC721Received(address,uint256,bytes) (contracts/trait/ERC721INFT.sol#438-459) uses assembly
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-use

Address.functionCall(address,bytes) (contracts/lib/Address.sol#79-91) is never used and should be removed
Address.functionCall(address,bytes,string) (contracts/lib/Address.sol#95-95) is never used and should be removed
Address.functionCallWithValue(address,bytes,int256) (contracts/lib/Address.sol#105-114) is never used and should be removed
Message.functionStaticCall(address,bytes) (contracts/lib/Address.sol#115-116) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/lib/Address.sol#151-160) is never used and should be removed
Context.functionValue(address) (contracts/trait/Context.sol#147-151) is never used and should be removed
Context.functionValue(address,bytes) (contracts/trait/Context.sol#147-151) is never used and should be removed
ERC1822UPRS.authorizeUpgrade(address) (contracts/trait/ERC1822UPRS.sol#140) is never used and should be removed
ERC1967Upgrade._changeAdmin(address) (contracts/trait/ERC1967Upgrade.sol#141-142) is never used and should be removed
ERC1967Upgrade._getBeacon() (contracts/trait/ERC1967Upgrade.sol#146-147) is never used and should be removed
ERC1967Upgrade._setAdmin(address) (contracts/trait/ERC1967Upgrade.sol#148-149) is never used and should be removed
ERC1967Upgrade._upgradeTo(neuImplementation) (contracts/trait/ERC1967Upgrade.sol#150-151) is never used and should be removed
ERC1967Upgrade._upgradeTo(neuImplementation,bytes) (contracts/trait/ERC1967Upgrade.sol#152-152) is never used and should be removed
EnumerableSet.set(EnumerableSet.Bytes32Set,uint256) (contracts/lib/EnumerableSet.sol#246-248) is never used and should be removed
EnumerableSet.set(EnumerableSet.UintSet,uint128) (contracts/lib/EnumerableSet.sol#249-250) is never used and should be removed
EnumerableSet.add(EnumerableSet.UintSet,uint128) (contracts/lib/EnumerableSet.sol#254-256) is never used and should be removed
EnumerableSet.set(EnumerableSet.Bytes32Set,uint256) (contracts/lib/EnumerableSet.sol#320-322) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet,uint128) (contracts/lib/EnumerableSet.sol#366-368) is never used and should be removed
EnumerableSet.setValues(EnumerableSet.AddressSet) (contracts/lib/EnumerableSet.sol#273-282) is never used and should be removed
EnumerableSet.setValues(EnumerableSet.UintSet) (contracts/lib/EnumerableSet.sol#294-303) is never used and should be removed
EnumerableSet.setValues(EnumerableSet.UintSet) (contracts/lib/EnumerableSet.sol#334-335) is never used and should be removed
StorageSlot.getBoolStorageSlot(bytes32) (contracts/lib/StorageSlot.sol#65-73) is never used and should be removed
StorageSlot.getBoolStorageSlot(bytes32) (contracts/lib/StorageSlot.sol#78-82) is never used and should be removed
String.toHexString(string,uint256) (contracts/lib/String.sol#19-50) is never used and should be removed
String.toHexString(string,uint256) (contracts/lib/String.sol#19-50) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#read-code

Variable Context._trustedForwarder (contracts/trait/Context.sol#1) is too similar to RarioEthereumToken.initialize(address,address,address,string,string).trustedForwarder_ (contracts/RarioEthereumToken.sol#15)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

ERC721INFT_E_NFT_TRANSFER_TO_NON_RECEIVER (contracts/trait/ERC721INFT.sol#80) is never used in RarioEthereumToken (contracts/RarioEthereumToken.sol#13-25)
RarioEthereumToken._get (contracts/trait/ERC1967Upgrade.sol#79) is never used in RarioEthereumToken (contracts/RarioEthereumToken.sol#13-25)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-call

```

```

RarioMarketplace.sol
ERC1967Upgrade.functionDelegateCall(address,bytes) (contracts/trait/ERC1967Upgrade.sol#200-206) uses delegatecall to a input-controlled function id
  - (success,returnData) = target.delegatecall(data) (contracts/trait/ERC1967Upgrade.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

Reentrancy in Marketplace._distributeFunds(address,string,uint256) (contracts/trait/Marketplace.sol#596-615):
  External calls:
    - _SendValue(treasury,platformPayableAccount,gasLimitMedium) (contracts/trait/Marketplace.sol#607)
      - (success) = user.call(gas,gasLimit,value:amount) (contracts/trait/Marketplace.sol#627)
        - _SendValue(treasury,platformPayableAccount,gasLimitMedium) (contracts/trait/Marketplace.sol#627)
      - (success) = user.call(gas,gasLimit,value:amount) (contracts/trait/Marketplace.sol#627)
    State variables written after the call(s):
      - _SendValue(treasury,platformPayableAccount,gasLimitMedium) (contracts/trait/Marketplace.sol#609)
        - pendingPlatformWithdrawals[user] += amount (contracts/trait/Marketplace.sol#632)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities

Marketplace.placeBid(uint256,address,string,uint256) (contracts/trait/Marketplace.sol#297-377) ignores return value by IERC20(currencyContractAddress).transferFrom(msgSender(),address(this),bidValue) (contracts/trait/Marketplace.sol#377)
Marketplace._distributeFunds(address,uint256) (contracts/trait/Marketplace.sol#596-615) ignores return value by IERC20(contractAddress).transferFrom(msgSender(),_treasury,platformPayableAmount) (contracts/trait/Marketplace.sol#611)
Marketplace._distributeFunds(address,uint256) (contracts/trait/Marketplace.sol#596-615) ignores return value by IERC20(contractAddress).transferFrom(msgSender(),seller,sellerPayableAmount) (contracts/trait/Marketplace.sol#612)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer

Marketplace.cancelBid(uint256) (contracts/trait/Marketplace.sol#155-156) is an upgradeable contract that does not protect its initialize functions: RarioMarketplace.initialize(address,address,address) (contracts/RarioMarketplace.sol#17-23). Anyone can de let the contract with: ERC1822UPRS.upgradeTo(address) (contracts/trait/ERC1822UPRS.sol#32-35)(ERC1822UPRS.upgradeToAndCall(address,bytes)) (contracts/trait/ERC1822UPRS.sol#45-48) Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-upgradeable-contract

Marketplace._cancelListing(uint256) (contracts/trait/Marketplace.sol#550-563) uses a dangerous strict equality:
  - require(bool,string)(!isValidSeller(msgSender()),'ERR_UNSUPPORTED_TOKEN_WALLET') (contracts/trait/Marketplace.sol#553)
Marketplace._cancelListing(uint256) (contracts/trait/Marketplace.sol#550-563) uses a dangerous strict equality:
  - require(bool,string)(listing.endTime == 0,'ERR_AUCTION_IN_PROGRESS') (contracts/trait/Marketplace.sol#554)
Marketplace._cancelListing(uint256) (contracts/trait/Marketplace.sol#550-563) uses a dangerous strict equality:
  - currencyType == 1 (contracts/trait/Marketplace.sol#560)
Marketplace._sendValue(address,uint256,uint256) (contracts/trait/Marketplace.sol#17-634) uses a dangerous strict equality:
  - amount == value (contracts/trait/Marketplace.sol#1862)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in Marketplace._createListing(uint256,uint256,string,address,address,address,uint16) (contracts/trait/Marketplace.sol#462-490):
  External calls:
    - require(bool,string)(isValidSeller(msgSender()),'ERR_UNSUPPORTED_TOKEN_WALLET') (contracts/trait/Marketplace.sol#459)
    - _TransferToken(listing.seller,listing.buyer,listing.listingId,listing.currencyType) (contracts/trait/Marketplace.sol#460)
      - IERC721(_trustedForwarder_,address(this)).transferFrom(from,to,to tokenId_) (contracts/trait/Marketplace.sol#513-515)
    State variables written after the call(s):
      - listing.currencyType = currencyType (contracts/trait/Marketplace.sol#461)
Reentrancy in Marketplace.placeBid(uint256,address,string,uint256) (contracts/trait/Marketplace.sol#297-377):

```

```

External calls:
- requireBool(string) (_isValidWallet(buyerTokenWallet),ERR_UNSUPPORTED_TOKEN_WALLET) (contracts/traits/Marketplace.sol#306)
State variables written after the calls:
- _listing.endTime = block.timestamp (contracts/traits/Marketplace.sol#324)
- listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#325)
- listing.buyerPaymentAddress = address(buyerPaymentAddress) (contracts/traits/Marketplace.sol#326)
- listing.bidCurrency = currency (contracts/traits/Marketplace.sol#327)
- listing.bidAmount = bidValue (contracts/traits/Marketplace.sol#328)
- listing.listingPrice = listingPrice (contracts/traits/Marketplace.sol#329)
- listing.buyerPaymentWallet = address(msgSender()) (contracts/traits/Marketplace.sol#330)
- listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#339)
- listing.bidAmount = bidValue (contracts/traits/Marketplace.sol#340)
- listing.endTime = block.timestamp + listing.duration (contracts/traits/Marketplace.sol#343)
- listing.bidAmount = bidValue (contracts/traits/Marketplace.sol#345)
- listing.buyerPaymentAddress = address(buyerPaymentAddress) (contracts/traits/Marketplace.sol#350)
- listing.buyerPaymentWallet = address(msgSender()) (contracts/traits/Marketplace.sol#351)
- listing.endTime = block.timestamp + listing.durationOnDuration (contracts/traits/Marketplace.sol#366)
References: https://github.com/crytic/solcifer/wiki/Detector-Documentation#entrancy-vulnerabilities-1

Marketplace._isValidWallet(address) _isValid (contracts/traits/Marketplace.sol#501) is a local variable never initialized
Reference: https://github.com/crytic/solcifer/wiki/Detector-Documentation#uninitialized-local-variables

AccessControl._grantRole(bytes32,address) (contracts/traits/AccessControl.sol#227-233) ignores return value by _roleMembers[role].add(account) (contracts/traits/AccessControl.sol#230)
AccessControl._revokeRole(bytes32,address) (contracts/traits/AccessControl.sol#235-241) ignores return value by _roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#238)
Reference: https://github.com/crytic/solcifer/wiki/Detector-Documentation#unused-return

Variable `Marketplace._isValidWallet(address)` isValid (contracts/traits/Marketplace.sol#501) in Marketplace._isValidWallet(address) (contracts/traits/Marketplace.sol#500-506) potentially used before declaration: isVali (contracts/traits/Marketplace.sol#500-506)
Reference: https://github.com/crytic/solcifer/wiki/Detector-Documentation#pre-declaration-use-of-local-variables

Reentrancy in Marketplace._createListing(uint256,uint256,string,address,address,address,uint8) (contracts/traits/Marketplace.sol#442-498):
External calls:
- requireBool(string) (_isValid(wallet,msgSender()),ERR_UNSUPPORTED_TOKEN_WALLET) (contracts/traits/Marketplace.sol#459)
- _transferFrom(wallet,config.getAllowExternalWallets()) (contracts/traits/Marketplace.sol#460)
- _transferFrom(wallet,config.getAllowInternalMarketplace()) (contracts/traits/Marketplace.sol#461)
- IERC21(trustedToken).transferFrom(from,to,tokeId) (contracts/traits/Marketplace.sol#515-519)
State variables written after the calls():
- _listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#445)
- _listing.buyerPaymentAddress = address(buyerPaymentAddress) (contracts/traits/Marketplace.sol#446)
Reentrancy in Marketplace._reloadConfiguration() (contracts/traits/Marketplace.sol#423-440):
External calls:
- config.getGasLimitMedium(gasLimitMedium) = config.getGasLimits() (contracts/traits/Marketplace.sol#458)
State variables written after the calls():
- _allowExternalWallets = config.getAllowExternalWallets() (contracts/traits/Marketplace.sol#455)
- _allowInternalMarketplace = config.getAllowInternalMarketplace() (contracts/traits/Marketplace.sol#456)
- auctionIncrementInBasisPoints = config.getAuctionIncrementInBasisPoints() (contracts/traits/Marketplace.sol#453)
- _listingBuyin = config.getListingBuyin() (contracts/traits/Marketplace.sol#451)
- _listingCurrency = currencyManager.getListingCurrency() (contracts/traits/Marketplace.sol#457)
- _platformFee = currencyManager.getPlatformFee() (contracts/traits/Marketplace.sol#458)
- delete supportedCurrencies (contracts/traits/Marketplace.sol#458)
- supportedCurrencies = currencyManager.supportedCurrencies() (contracts/traits/Marketplace.sol#459)
Reentrancy in Marketplace._sendAndValidate(uint256,uint256,uint256) (contracts/traits/Marketplace.sol#647-648):
External calls:
- (success,user.callgas(gasLimit,value:amount)) (contracts/traits/Marketplace.sol#647)
State variables written after the calls():
- _pendingWithdrawalUser = pendingWithdrawalsUser + amount (contracts/traits/Marketplace.sol#632)
Reentrancy in Marketplace._placeBid(uint256,address,string,uint256) (contracts/traits/Marketplace.sol#597-597):
External calls:
- requireBool(string) (_isValid(wallet,buyerTokenWallet),ERR_UNSUPPORTED_TOKEN_WALLET) (contracts/traits/Marketplace.sol#501-505)
- config._isValid(wallet) (contracts/traits/Marketplace.sol#501-505)
- _finaliseBid(listingId,tokeId,listingBuyin,listingCurrency,listing.bidAmount) (contracts/traits/Marketplace.sol#611)
- IERC21(trustedToken).transferFrom(from,to,tokeId) (contracts/traits/Marketplace.sol#611)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(seller,sellerPayableAmount) (contracts/traits/Marketplace.sol#612)
External calls sending eth:
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#529)
- requireBool(string) (_isValid(wallet),gasLimit,value:amount) (contracts/traits/Marketplace.sol#627)
State variables written after the calls():
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#529)
- _finaliseListings(listingId) (contracts/traits/Marketplace.sol#529)
Reentrancy in Marketplace._placeBid(uint256,address,uint256) (contracts/traits/Marketplace.sol#546-547):
External calls:
- _relodConfiguration() (contracts/traits/Marketplace.sol#546)
- config.getGasLimitMedium(gasLimitMedium) = config.getGasLimits() (contracts/traits/Marketplace.sol#428)
State variables written after the calls():
- _hasConfiguration = true (contracts/traits/Marketplace.sol#547)
Reference: https://github.com/crytic/solcifer/wiki/Detector-Documentation#entrancy-vulnerabilities-2

Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#550-563):
External calls:
- IERC21(trustedToken).transferFrom(address(this),listing.sellerTokenWallet,tokeId) (contracts/traits/Marketplace.sol#560)
Event emitted after the call():
- listing.sellerTokenWallet = address(sellerTokenWallet) (contracts/traits/Marketplace.sol#562)
Reentrancy in Marketplace._cancelListing(uint256,uint256,uint256,address,address,uint8) (contracts/traits/Marketplace.sol#442-498):
External calls:
- requireBool(string) (_isValid(wallet,msgSender()),ERR_UNSUPPORTED_TOKEN_WALLET) (contracts/traits/Marketplace.sol#459)
- config._isValid(wallet) (contracts/traits/Marketplace.sol#501-505)
- _transferFrom(wallet,config.getAllowInternalMarketplace()) (contracts/traits/Marketplace.sol#460)
- IERC21(trustedToken).transferFrom(from,to,tokeId) (contracts/traits/Marketplace.sol#515-519)
Event emitted after the call():
- listingCreated(listingType,tokeId,msgSender(),sellerPaymentWallet,buyerTokenWallet,buyerPaymentWallet,listing.price,listing.duration,listing.extensionDuration,listing.endTime) (contracts/traits/Marketplace.sol#445-498)
Reentrancy in Marketplace._finaliseListing(uint256) (contracts/traits/Marketplace.sol#565-594):
External calls:
- _transferToken(address(this),listing.buyerTokenWallet,listing.tokeId,listing.currency,listing.bidAmount) (contracts/traits/Marketplace.sol#576)
- (platformFee,sellerPayableAmount) = distributeFunds(listing.sellerPaymentWallet,listing.bidCurrency,listing.bidAmount) (contracts/traits/Marketplace.sol#577-581)
- (success) = user.callgas(gasLimit,value:amount) (contracts/traits/Marketplace.sol#607)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(seller,sellerPayableAmount) (contracts/traits/Marketplace.sol#612)
External calls sending eth:
- (platformFee,sellerPayableAmount) = distributeFunds(listing.sellerPaymentWallet,listing.bidCurrency,listing.bidAmount) (contracts/traits/Marketplace.sol#577-581)
Event emitted after the call():
- (success) = user.callgas(gasLimit,value:amount) (contracts/traits/Marketplace.sol#607)
Event emitted after the call():
- Upgraded(newImplementation) (contracts/traits/ERC1967Upgrade.sol#158)
Reentrancy in Marketplace._finaliseListing(uint256) (contracts/traits/Marketplace.sol#207-377):
External calls:
- requireBool(string) (_isValid(wallet,tokeId),ERR_UNSUPPORTED_TOKEN_WALLET) (contracts/traits/Marketplace.sol#306)
- config._isValid(wallet) (contracts/traits/Marketplace.sol#501-505)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#325)
- IERC21(trustedToken).transferFrom(from,to,tokeId) (contracts/traits/Marketplace.sol#515-519)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(msgSender(),treasury,platformFeePayableAmount) (contracts/traits/Marketplace.sol#611)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(msgSender()),seller,sellerPayableAmount) (contracts/traits/Marketplace.sol#612)
External calls:
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#329)
- (success) = user.callgas(gasLimit,value:amount) (contracts/traits/Marketplace.sol#627)
Event emitted after the call():
- ListingFinalised(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256) (contracts/traits/Marketplace.sol#533-559)
Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#595-619):
External calls:
- _transferToken(address(this),listing.buyerTokenWallet,listing.tokeId,listing.currency,listing.bidAmount) (contracts/traits/Marketplace.sol#576)
- (platformFee,sellerPayableAmount) = distributeFunds(listing.sellerPaymentWallet,listing.bidCurrency,listing.bidAmount) (contracts/traits/Marketplace.sol#577-581)
- (success) = user.callgas(gasLimit,value:amount) (contracts/traits/Marketplace.sol#607)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(seller,sellerPayableAmount) (contracts/traits/Marketplace.sol#612)
Event emitted after the call():
- listingCancelled(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256) (contracts/traits/Marketplace.sol#533-559)
Reentrancy in Marketplace._placeBid(uint256,address,uint256,uint256) (contracts/traits/Marketplace.sol#207-377):
External calls:
- requireBool(string) (_isValid(wallet,buyerTokenWallet),ERR_UNSUPPORTED_TOKEN_WALLET) (contracts/traits/Marketplace.sol#306)
- config._isValid(wallet) (contracts/traits/Marketplace.sol#501-505)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#325)
- IERC21(trustedToken).transferFrom(from,to,tokeId) (contracts/traits/Marketplace.sol#515-519)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(msgSender(),treasury,platformFeePayableAmount) (contracts/traits/Marketplace.sol#611)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(msgSender()),seller,sellerPayableAmount) (contracts/traits/Marketplace.sol#612)
External calls:
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#329)
- (success) = user.callgas(gasLimit,value:amount) (contracts/traits/Marketplace.sol#627)
Event emitted after the call():
- ListingFinalised(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256) (contracts/traits/Marketplace.sol#533-559)
Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#595-619):
External calls:
- requireBool(string) (_isValid(wallet,tokeId),ERR_UNSUPPORTED_TOKEN_WALLET) (contracts/traits/Marketplace.sol#306)
- config._isValid(wallet) (contracts/traits/Marketplace.sol#501-505)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#325)
- IERC21(trustedToken).transferFrom(from,to,tokeId) (contracts/traits/Marketplace.sol#515-519)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(msgSender(),treasury,platformFeePayableAmount) (contracts/traits/Marketplace.sol#611)
- IERC20(config.getAllowExternalMarketplace()).transferFrom(msgSender()),seller,sellerPayableAmount) (contracts/traits/Marketplace.sol#612)
Event emitted after the call():
- ListingCancelled(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256) (contracts/traits/Marketplace.sol#533-559)
Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#401-407):
External calls:
- _sendValue(amount,gasLimit) (contracts/traits/Marketplace.sol#405)
- (success) = user.callgas(gasLimit,value:amount) (contracts/traits/Marketplace.sol#627)
Event emitted after the call():
- Withdrawn(amount) (contracts/traits/Marketplace.sol#406)
Reference: https://github.com/crytic/solcifer/wiki/Detector-Documentation#entrancy-vulnerabilities-3

```

RarioMarketplaceConfig.sol

ERC1967Upgrade, functionDelegateCall(address,bytes) (contracts/traits/ERC1967Upgrade.sol#0x120-0x206) uses delegatecall to a input-controlled function id
- (successes,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#0x20-0x204)
Reference: <https://github.com/crytic/slither/v1k/Detector-Documentation#controlled-delegatecall>

RarioMarketplaceConfig (contracts/RarioMarketplaceConfig.sol#1-23) is an upgradeable contract that does not protect its initialize functions: RarioMarketplaceConfig.initialize(address,address,address) (contracts/RarioMarketplaceConfig.Config.sol#1-23). Anyone can delete the contract with ERC1822Upgrade.toAddress() (contracts/traits/ERC1822UPGRADE.sol#32-35)ERC1822UPGRADE.upgradeToAndCall(address,bytes) (contracts/traits/ERC1822UPGRADE.sol#45-48) Reference: <https://github.com/crytic/slither/v1k/Detector-Documentation#upgradeable-upgradeable-contract>

MarketplaceConfig_existsUserWallet(address)_sValue (contracts/traits/MarketplaceConfig.sol#145) is a local variable never initialized
Reference: <https://github.com/crytic/slither/v1k/Detector-Documentation#uninitialised-local-variables>

AccessControl_greanRole(bytes2,address) (contracts/traits/AccessControl.sol#227-233) ignores return value by _roleMembers[role].add(account) (contracts/traits/AccessControl.sol#230)
AccessControl_revocateRole(bytes2,address) (contracts/traits/AccessControl.sol#235-241) ignores return value by _roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#238)
MarketplaceConfig_existsUserWallet(address) (contracts/traits/MarketplaceConfig.sol#144-150) ignores return value by IUserDB(rarioUserDB).existsUserWallet(wallet) (contracts/traits/MarketplaceConfig.sol#145-149)
Reference: <https://github.com/crytic/slither/v1k/Detector-Documentation#access-control>

Value_MarketplaceConfig_existsUserWallet(address)_sValue (contracts/traits/MarketplaceConfig.sol#145) in MarketplaceConfig._existsUserWallet(address) (contracts/traits/MarketplaceConfig.sol#144-150) potentially used before declare
Init event = UserEvent("existsUserWallet(address)",MarketplaceConfig.sol#144-150)
Reference: <https://github.com/crytic/slither/v1k/Detector-Documentation#declaration-use-of-local-variables>

Reentrancy in ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#92-107):
- External call:
- _functionDelegateCall(newImplementation,data) (contracts/traits/ERC1967Upgrade.sol#92)
- (successes,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#92-0x200)
- _functionDelegateCall(newImplementation,data) (contracts/traits/ERC1967Upgrade.sol#92-0x200)
- (successes,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
Event emitted after the call(s):
- UpgradedContract("newImplementation",ContractAddress,bytes) (contracts/traits/ERC1967Upgrade.sol#105)
- upgradedTo(newImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
Reference: <https://github.com/crytic/slither/v1k/Detector-Documentation#reentrancy-vulnerabilities-3>

Address_isContract(address) (contracts/lib/Address.sol#2-36) user assembly
- INLINE ASM (contracts/lib/Address.sol#32-34)
Address_verifyContractAddress(bytes,string) (contracts/lib/Address.sol#165-168) user assembly
- INLINE ASM (contracts/lib/Address.sol#165-168)
EnumerableSet_value(EnumerableSet,AddressSet) (contracts/lib/EnumerableSet.sol#273-282) uses assembly
- INLINE ASM (contracts/lib/EnumerableSet.sol#277-279)
EnumerableSet_setValue(EnumerableSet,AddressSet,Address) (contracts/lib/EnumerableSet.sol#346-355) uses assembly
- INLINE ASM (contracts/lib/EnumerableSet.sol#350-352)
StorageSlot_getAddressSlot(bytes32) (contracts/lib/StorageSlot.sol#5-55) uses assembly
- INLINE ASM (contracts/lib/StorageSlot.sol#5-55)
StorageSlot_getStorageSlot(bytes32) (contracts/lib/StorageSlot.sol#59-64) uses assembly
- INLINE ASM (contracts/lib/StorageSlot.sol#61-64)
StorageSlot_getByte32Slot(bytes32) (contracts/lib/StorageSlot.sol#67-73) uses assembly
- INLINE ASM (contracts/lib/StorageSlot.sol#67-73)
StorageSlot_getInt32Slot(bytes32) (contracts/lib/StorageSlot.sol#77-82) uses assembly
- INLINE ASM (contracts/lib/StorageSlot.sol#77-82)
Context_newContext() (contracts/lib/Context.sol#19-21) uses assembly
- INLINE ASM (contracts/lib/Context.sol#19-21)
Reference: <https://github.com/crytic/slither/v1k/Detector-Documentation#assembly-use>

RarioUsers.sol

```

AccessControl_Upgrade_.functionDelegateCall(address,bytes) (contracts/traits/AccessControl.sol#200-206) uses delegatecall to a input-controlled function id
- (success, returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#20)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

RarioUsers (contracts/RarioUsers.sol#1-23) ignores return value by _roleMembers(role).add(account) (contracts/traits/AccessControl.sol#230)
AccessControl_.revokeRole(bytes32,address) (contracts/traits/AccessControl.sol#233-241) ignores return value by _roleMembers(role).remove(account) (contracts/traits/AccessControl.sol#238)
UserDB.addUser(uint256,address) (contracts/traits/UserDB.sol#4-15) ignores return value by rarioIds.add(rarioId) (contracts/traits/UserDB.sol#4)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Reentrancy in ERC1967Upgrade_.upgradeToAndCall(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#82-107):
External call to newImplementation.data (contracts/traits/ERC1967Upgrade.sol#43)
- _functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address)",oldImplementation)) (contracts/traits/ERC1967Upgrade.sol#20)
- _functionDelegateCall(newImplementation,abi.encodeWithSignature("oldImplementation()")) (contracts/traits/ERC1967Upgrade.sol#100)
Event emitted after the call(s):
- Upgraded(newImplementation) (contracts/traits/ERC1967Upgrade.sol#55)
- Upgraded(oldImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.iContext(address) (contracts/lib/Address.sol#45-56) uses assembly
- INLINE ASM (contract/lib/Address.sol#45-54)
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#168-188) uses assembly
EnumerableSet.values(EnumerableSet.AddressSet) (contracts/lib/EnumerableSet.sol#273-282) uses assembly
- INLINE ASM (contract/lib/EnumerableSet.sol#277-279)
EnumerableSet.contains(EnumerableSet.AddressSet,uint256) (contracts/lib/EnumerableSet.sol#346-355) uses assembly
- INLINE ASM (contract/lib/EnumerableSet.sol#350-352)
StorageSlot.getAddressSlot(bytes32) (contract/lib/StorageSlot.sol#1-55) uses assembly
- INLINE ASM (contract/lib/StorageSlot.sol#50-54)
StorageSlot.getUint256Slot(bytes32) (contract/lib/StorageSlot.sol#1-55)
- INLINE ASM (contract/lib/StorageSlot.sol#46-49)
StorageSlot.getBoolSlot(bytes32) (contract/lib/StorageSlot.sol#49-53) uses assembly
StorageSlot.getUnit256Slot(bytes32) (contract/lib/StorageSlot.sol#178-22)
- INLINE ASM (contract/lib/StorageSlot.sol#78-81)
Context.setAdmin(address) (contracts/traits/Context.sol#19-21)
- INLINE ASM (contract/traits/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Address.functionCall(address,bytes) (contracts/lib/Address.sol#79-81) is never used and should be removed
Address.functionCall(address,bytes,string) (contracts/lib/Address.sol#89-95) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contract/lib/Address.sol#105-114) is never used and should be removed
Address.functionStaticCall(addresses,bytes) (contract/lib/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(addresses,bytes,string) (contract/lib/Address.sol#145-160) is never used and should be removed
Address.functionStaticCall(addresses,bytes,bytes) (contract/lib/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(addresses,bytes,bytes,string) (contract/lib/Address.sol#145-160) is never used and should be removed
Context.setMsgData() (contracts/traits/Context.sol#27-33) is never used and should be removed
ERC165Interface.initializeERC165Interface() (contract/traits/ERC165Interface.sol#182-22) is never used and should be removed
ERC1967Upgrade_.autoUpgrade(address) (contracts/traits/ERC1967Upgrade.sol#11-14) is never used and should be removed
ERC1967Upgrade_.getAdmin() (contracts/traits/ERC1967Upgrade.sol#124-126) is never used and should be removed
ERC1967Upgrade_.getBeacon() (contracts/traits/ERC1967Upgrade.sol#131-133) is never used and should be removed
ERC1967Upgrade_.getBeacon() (contracts/traits/ERC1967Upgrade.sol#174) is never used and should be removed
ERC1967Upgrade_.upgradeBeaconAtSlot(address,bytes,bytes,bytes) (contract/traits/ERC1967Upgrade.sol#189) is never used and should be removed
EnumerableSet.set(EnumerableSet.AddressSet) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.Values(EnumerableSet.AddressSet) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#157-159) is never used and should be removed
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#157-159) is never used and should be removed
EnumerableSet.contains(EnumerableSet.AddressSet,address) (contract/lib/EnumerableSet.sol#240-242) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#167-169) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#106-109) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#207-209) is never used and should be removed
EnumerableSet.Values(EnumerableSet.UintSet) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
StorageSlot.getUnit256Slot(bytes32) (contract/lib/StorageSlot.sol#178-22) is never used and should be removed
String.equals(string,string) (contracts/lib/Strings.sol#67-71) is never used and should be removed
String.toHexString(uint256) (contracts/lib/Strings.sol#141-144) is never used and should be removed
String.toPaddedString(uint256) (contracts/lib/Strings.sol#141-144) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

```

traits/AccessControl.sol

```

AccessControl_.grantRole(bytes32,address) (contracts/traits/AccessControl.sol#227-233) ignores return value by _roleMembers(role).add(account) (contracts/traits/AccessControl.sol#230)
AccessControl_. revokeRole(bytes32,address) (contracts/traits/AccessControl.sol#235-241) ignores return value by _roleMembers(role).remove(account) (contracts/traits/AccessControl.sol#238)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

EnumerableSet.values(EnumerableSet.AddressSet) (contracts/lib/EnumerableSet.sol#273-282) uses assembly
EnumerableSet.values(EnumerableSet.UintSet) (contract/lib/EnumerableSet.sol#346-355) uses assembly
- INLINE ASM (contract/lib/EnumerableSet.sol#350-352)
Context.setAdmin(address) (contracts/traits/Context.sol#19-21)
- INLINE ASM (contract/traits/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

AccessControl_.setRoleAdmin(bytes32,bytes32) (contracts/traits/AccessControl.sol#222-225) is never used and should be removed
AccessControl_.setupRole(bytes32,address) (contracts/traits/AccessControl.sol#212-214) is never used and should be removed
AccessControl_.setRoleAdmin(bytes32,bytes32) (contract/traits/AccessControl.sol#212-214) is never used and should be removed
Context.setRoleAdmin(bytes32,bytes32) (contracts/traits/Context.sol#27-33) is never used and should be removed
Context.setRoleAdmin(bytes32,bytes32) (contract/traits/Context.sol#27-33) is never used and should be removed
Context.setRoleAdmin(bytes32,bytes32) (contract/traits/AccessControl.sol#212-214) is never used and should be removed
Context.setRoleAdmin(bytes32,bytes32) (contract/traits/AccessControl.sol#212-214) is never used and should be removed
EnumerableSet._values(EnumerableSet.Set) (contracts/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet._values(EnumerableSet.UintSet) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#157-159) is never used and should be removed
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#157-159) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.set(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.set(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#167-169) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#106-109) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (contract/lib/EnumerableSet.sol#207-209) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet, uint256) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.set(EnumerableSet.UintSet, uint256) (contract/lib/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.set(EnumerableSet.UintSet, uint256) (contract/lib/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet, uint256) (contract/lib/EnumerableSet.sol#167-169) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet, uint256) (contract/lib/EnumerableSet.sol#106-109) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet, uint256) (contract/lib/EnumerableSet.sol#207-209) is never used and should be removed
EnumerableSet.contains(EnumerableSet.AddressSet) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.contains(EnumerableSet.AddressSet) (contract/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.set(EnumerableSet.AddressSet, address) (contract/lib/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.set(EnumerableSet.AddressSet, address) (contract/lib/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.remove(EnumerableSet.AddressSet, address) (contract/lib/EnumerableSet.sol#167-169) is never used and should be removed
EnumerableSet.remove(EnumerableSet.AddressSet, address) (contract/lib/EnumerableSet.sol#106-109) is never used and should be removed
EnumerableSet.remove(EnumerableSet.AddressSet, address) (contract/lib/EnumerableSet.sol#207-209) is never used and should be removed
String.equals(string,string) (contracts/lib/Strings.sol#67-71) is never used and should be removed
String.toHexString(uint256) (contracts/lib/Strings.sol#141-144) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

```

traits/Context.sol

```

Context.setMsgData() (contracts/traits/Context.sol#1-21)
- INLINE ASM (contract/traits/Context.sol#1-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Context.refusedForwarder(address) (contracts/traits/Context.sol#12-14) is never used and should be removed
Context.setMsgData() (contracts/traits/Context.sol#27-33) is never used and should be removed
Context.setMsgData() (contracts/traits/Context.sol#141-143) is never used and should be removed
Context.setMsgData() (contracts/traits/Context.sol#141-143) is never used and should be removed
Context.setMsgData() (contracts/traits/Context.sol#141-143) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

```

traits/ERC165Interface.sol

`ERC165Interface.initialize(ERC165Interface)` (contracts/traits/ERC165Interface.sol#20-22) is never used and should be removed
`ERC165Interface.registerInterface(bytes4)` (contracts/traits/ERC165Interface.sol#42-45) is never used and should be removed
Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code>

```

traits/CurrencyManager.sol
ERC1967Upgrade.functionDelegateCall(address,bytes) (contracts/traits/ERC1967Upgrade.sol#200-202) uses delegatecall to a input-controlled function id
  - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#200)
References: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

CurrencyManager._convert(string,string,uint256) (contracts/traits/CurrencyManager.sol#34) is a local variable never initialized
CurrencyManager._convert(string,string,uint256).currencyPairDecimals (contracts/traits/CurrencyManager.sol#34) is a local variable never initialized
References: https://github.com/crytic/slither/wiki/Detector-Documentation#initialized-local-variables

AccessControl.grantRole(bytes32,address) (contracts/traits/AccessControl.sol#227-233) ignores return value by roleMembers[role].add(account) (contracts/traits/AccessControl.sol#230)
AccessControl.revokeRole(bytes32,address) (contracts/traits/AccessControl.sol#234-236) ignores return value by roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#233)
CurrencyManager._convert(string,string,uint256) (contracts/traits/CurrencyManager.sol#325-327) ignores return value by currencyPairs[symbol].add(pairSymbol) (contracts/traits/CurrencyManager.sol#31)
CurrencyManager._convert(string,string,uint256) (contracts/traits/CurrencyManager.sol#328-344) ignores return value by IFixedFeeContractAddress().latestRole() (contracts/traits/CurrencyManager.sol#345-346)
CurrencyManager._convert(string,string,uint256) (contracts/traits/CurrencyManager.sol#347-353) ignores return value by currencyPairs[feeContractAddress].add(pairSymbol) (contracts/traits/CurrencyManager.sol#350)
CurrencyManager._removeCurrency(string) (contracts/traits/CurrencyManager.sol#345-355) ignores return value by symbols.remove(symbol) (contracts/traits/CurrencyManager.sol#352)
CurrencyManager._removeCurrency(string) (contracts/traits/CurrencyManager.sol#357-419) ignores return value by currencyPairs[symbol].remove(symbol) (contracts/traits/CurrencyManager.sol#400)
References: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Variable.CurrencyManager._convert(string,string,uint256).currencyPairDecimals (contracts/traits/CurrencyManager.sol#34) is in CurrencyManager._convert(string,string,uint256) (contracts/traits/CurrencyManager.sol#328-344) potentially used before declaration
Variable.CurrencyManager._convert(string,string,uint256).price (contracts/traits/CurrencyManager.sol#347) is in CurrencyManager._convert(string,string,uint256) (contracts/traits/CurrencyManager.sol#328-344) potentially used before declaration
Variable.CurrencyManager._convert(string,string,uint256).symbol (contracts/traits/CurrencyManager.sol#347) is in CurrencyManager._convert(string,string,uint256) (contracts/traits/CurrencyManager.sol#328-344) potentially used before declaration
Variable.CurrencyManager._convert(string,string,uint256).symbol (contracts/traits/CurrencyManager.sol#354-363) is in CurrencyManager._convert(string,string,uint256) (contracts/traits/CurrencyManager.sol#345-360)
Variable.CurrencyManager._removeCurrency(string) (contracts/traits/CurrencyManager.sol#345-355) ignores return value by symbols.remove(symbol) (contracts/traits/CurrencyManager.sol#352)
Variable.CurrencyManager._removeCurrency(string) (contracts/traits/CurrencyManager.sol#357-419) ignores return value by currencyPairs[symbol].remove(symbol) (contracts/traits/CurrencyManager.sol#400)
References: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Reentrancy in ERC1967Upgrade._upgradeToAndCall(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#82):
  External calls:
    - _functionDelegateCall((newImplementation,data)) (contracts/traits/ERC1967Upgrade.sol#82)
      - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#82)
        - _functionImplementation(bytes20) (contracts/traits/ERC1967Upgrade.sol#82)
          - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#82)
    Event emitted after the call(s):
      - UpgradeTo(newImplementation) (contracts/traits/ERC1967Upgrade.sol#85)
        - _upgradeTo(newImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
References: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/lib/Address.sol#26-36) uses assembly
  - INLINE ASM (contract/lib/Address.sol#32-34)
Address.verifyCallResult(bytes4,address,bytes) (contracts/lib/Address.sol#16-18) uses assembly
  - INLINE ASM (contract/lib/Address.sol#16-18)
EnumerableSet.Values(EnumerableSet.AddressSet) (contracts/lib/EnumerableSet.sol#275-282) uses assembly
  - INLINE ASM (contract/lib/EnumerableSet.sol#277-279)
EnumerableSet.AddressSet(EnumerableSet.AddressSet) (contracts/lib/EnumerableSet.sol#346-355) uses assembly
  - INLINE ASM (contract/lib/EnumerableSet.sol#350-352)
Storage.getAddreses32(bytes32) (contracts/lib/Storageslot.sol#81-85) uses assembly
  - INLINE ASM (contract/lib/Storageslot.sol#81-85)
Storage.getBoolSlot(bytes32) (contracts/lib/Storageslot.sol#60-64) uses assembly
  - INLINE ASM (contract/lib/Storageslot.sol#64-65)
Storage.getUin256Slot(bytes32) (contracts/lib/Storageslot.sol#69-73) uses assembly
  - INLINE ASM (contract/lib/Storageslot.sol#70-73)
Storage.getUin256Slot(bytes2) (contract/lib/Storageslot.sol#78-82) uses assembly
  - INLINE ASM (contract/lib/Storageslot.sol#79-81)
Context._msgSender() (contracts/traits/Context.sol#16-18) uses assembly
  - INLINE ASM (contract/traits/Context.sol#19-21)
References: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-use

AccessControl.setup32(address) (contracts/traits/AccessControl.sol#212-214) is never used and should be removed
AccessControl.initializeAccessControl() (contracts/traits/AccessControl.sol#80-83) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/lib/Address.sol#105-110) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/lib/Address.sol#105-110) is never used and should be removed
Address.functionStaticCall(addresses,bytes) (contracts/lib/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(addresses,bytes) (contracts/lib/Address.sol#145-160) is never used and should be removed
Address.functionStaticCall(addresses,bytes) (contracts/lib/Address.sol#162-164) is never used and should be removed
Context._msgData() (contracts/traits/Context.sol#10-12) is never used and should be removed
Context._structuredForwarder(address) (contracts/traits/Context.sol#8-10) is never used and should be removed
Context._msgData() (contracts/traits/Context.sol#10-12) is never used and should be removed
Context._msgData() (contracts/traits/Context.sol#10-12) is never used and should be removed
ERC165Interface.initialize(ERC165Interface) (contracts/traits/ERC165Interface.sol#140-142) is never used and should be removed
ERC165Interface.registerInterface(bytes4) (contracts/traits/ERC165Interface.sol#142-145) is never used and should be removed
ERC165Upgrade.authorizeUpgrade(address) (contracts/traits/ERC165Upgrade.sol#40) is never used and should be removed
ERC165Upgrade._changeAdmin(address) (contracts/traits/ERC165Upgrade.sol#141-144) is never used and should be removed
ERC165Upgrade._getAdmin() (contracts/traits/ERC165Upgrade.sol#124-126) is never used and should be removed
ERC165Upgrade._setAdmin(address) (contracts/traits/ERC165Upgrade.sol#131-134) is never used and should be removed
ERC165Upgrade._setAdmin(address) (contracts/traits/ERC165Upgrade.sol#131-134) is never used and should be removed
ERC165Upgrade._upgradeToAndCall(address,bytes,bool) (contracts/traits/ERC165Upgrade.sol#66-70) is never used and should be removed
ERC165Upgrade.initialize(ERC165Upgrade) (contracts/traits/ERC165Upgrade.sol#15) is never used and should be removed
EnumerableSet.Add(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#181-185) is never used and should be removed
EnumerableSet.Add(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#181-185)
EnumerableSet.Uint32Set(uint256) (contracts/lib/EnumerableSet.sol#296-298) is never used and should be removed
EnumerableSet.Uint32Set(uint256) (contracts/lib/EnumerableSet.sol#296-298) is never used and should be removed
EnumerableSet.Uint32Set(uint256,bytes24) (contracts/lib/EnumerableSet.sol#356-358) is never used and should be removed
EnumerableSet.Contains(EnumerableSet.AddressSet,address) (contracts/lib/EnumerableSet.sol#240-242) is never used and should be removed
EnumerableSet.Contains(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#328-331) is never used and should be removed
EnumerableSet.Contains(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#328-331) is never used and should be removed
EnumerableSet.Length(EnumerableSet.Bytes32Set) (contracts/lib/EnumerableSet.sol#181-183) is never used and should be removed
EnumerableSet.Length(EnumerableSet.Bytes32Set) (contracts/lib/EnumerableSet.sol#181-183)
EnumerableSet.Remove(EnumerableSet.UintSet,uint256) (contracts/lib/EnumerableSet.sol#304-308) is never used and should be removed
EnumerableSet.Value(EnumerableSet.AddressSet) (contracts/lib/EnumerableSet.sol#273-282) is never used and should be removed
EnumerableSet.Value(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#346-355) is never used and should be removed
EnumerableSet.Value(EnumerableSet.UintSet) (contracts/lib/EnumerableSet.sol#240-242) is never used and should be removed
Faucable.initializeFaucable() (contracts/traits/Faucable.sol#30-32) is never used and should be removed
Storage.getUin256Slot(bytes32) (contracts/lib/Storageslot.sol#78-82) is never used and should be removed
String.equals(string,string) (contracts/lib/Strings.sol#7-73) is never used and should be removed
String.equals(string,string) (contracts/lib/Strings.sol#7-73) is never used and should be removed
String.equals(string,uint256) (contracts/lib/Strings.sol#14-34) is never used and should be removed
Upgradable.initializeUpgradable(address) (contracts/traits/Upgradable.sol#16-24) is never used and should be removed
References: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

```

```

traits/ERC1822UUPS.sol
ERC1967Upgrade.functionDelegateCall(address,bytes) (contracts/traits/ERC1967Upgrade.sol#200-206) uses delegatecall to a input-controlled function id
  - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

Reentrancy in ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#82-107):
  External calls:
    - _functionDelegateCall(newImplementation,data) (contracts/traits/ERC1967Upgrade.sol#82)
      - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
    - _functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address,bytes)",newImplementation)) (contracts/traits/ERC1967Upgrade.sol#100)
      - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
  Event emitted after the call(s):
    - Upgrade(neuImplementation) (contracts/traits/ERC1967Upgrade.sol#82-105)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/lib/Address.sol#26-36) uses assembly
  - INLINE ASM (contracts/lib/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#165-186) uses assembly
StorageSlot.getAddressSlot(bytes2) (contracts/lib/StorageSlot.sol#151-155) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#54-56)
StorageSlot.getAddressSlot(bytes2) (contracts/lib/StorageSlot.sol#1460-1464) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#44-46)
StorageSlot.getByte32Slot(bytes2) (contracts/lib/StorageSlot.sol#1469-1473) uses assembly
StorageSlot.getUin256Slot(bytes2) (contracts/lib/StorageSlot.sol#178-182) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#77-81)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Address.functionCall(addresses,bytes) (contracts/lib/Address.sol#79-81) is never used and should be removed
Address.functionCall(addresses,bytes,bytes) (contracts/lib/Address.sol#110-114) is never used and should be removed
Address.functionCallWithValue(addresses,bytes,uint256) (contracts/lib/Address.sol#122-133) is never used and should be removed
Address.functionStaticCall(addresses,bytes,string) (contracts/lib/Address.sol#141-153) is never used and should be removed
Address.functionStaticCall(addresses,bytes,bytes) (contracts/lib/Address.sol#145-149) is never used and should be removed
Address.functionStaticCall(addresses,bytes,bytes,bytes) (contracts/lib/Address.sol#155-160) is never used and should be removed
Address.functionStaticCall(addresses,bytes,bytes,bytes,bytes) (contracts/lib/Address.sol#162-167) is never used and should be removed
Address.getUin256Slot(bytes2) (contracts/traits/ERC1967Upgrade.sol#124-126) is never used and should be removed
ERC1967Upgrade.getAdmin() (contracts/traits/ERC1967Upgrade.sol#124-126) is never used and should be removed
ERC1967Upgrade.getBeacon() (contracts/traits/ERC1967Upgrade.sol#146-148) is never used and should be removed
ERC1967Upgrade.setBeacon(address) (contracts/traits/ERC1967Upgrade.sol#177) is never used and should be removed
ERC1967Upgrade.upgradeToAndCall(address,bytes,bytes) (contracts/traits/ERC1967Upgrade.sol#192-197) is never used and should be removed
ERC1967Upgrade.initializeERC1967Upgrade() (contracts/traits/ERC1967Upgrade.sol#198) is never used and should be removed
StorageSlot.getByte32Slot(bytes2) (contracts/lib/StorageSlot.sol#80-83) is never used and should be removed
StorageSlot.getUin256Slot(bytes2) (contracts/lib/StorageSlot.sol#84-87) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

ERC1822UUPS (contracts/traits/ERC1822UUPS.sol#20-69) does not implement functions:
  - ERC1822UUPS.authorizeUpgrade(address) (contracts/traits/ERC1822UUPS.sol#60)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions

```

```

traits/ERC1967Upgrade.sol
Reentrancy in ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#82-107):
  External calls:
    - _functionDelegateCall(newImplementation,data) (contracts/traits/ERC1967Upgrade.sol#82)
      - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
    - _functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address,bytes)",newImplementation)) (contracts/traits/ERC1967Upgrade.sol#100)
      - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
  Event emitted after the call(s):
    - Upgrade(neuImplementation) (contracts/traits/ERC1967Upgrade.sol#82-105)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/lib/Address.sol#26-36) uses assembly
  - INLINE ASM (contracts/lib/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#165-186) uses assembly
StorageSlot.getAddressSlot(bytes2) (contracts/lib/StorageSlot.sol#151-155) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#54-56)
StorageSlot.getAddressSlot(bytes2) (contracts/lib/StorageSlot.sol#1460-1464) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#44-46)
StorageSlot.getByte32Slot(bytes2) (contracts/lib/StorageSlot.sol#1469-1473) uses assembly
StorageSlot.getUin256Slot(bytes2) (contracts/lib/StorageSlot.sol#178-182) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#77-81)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Address.functionCall(addresses,bytes) (contracts/lib/Address.sol#79-81) is never used and should be removed
Address.functionCall(addresses,bytes,bytes) (contracts/lib/Address.sol#110-114) is never used and should be removed
Address.functionCallWithValue(addresses,bytes,uint256) (contracts/lib/Address.sol#122-133) is never used and should be removed
Address.functionStaticCall(addresses,bytes,string) (contracts/lib/Address.sol#141-153) is never used and should be removed
Address.functionStaticCall(addresses,bytes,bytes) (contracts/lib/Address.sol#145-149) is never used and should be removed
Address.functionStaticCall(addresses,bytes,bytes,bytes) (contracts/lib/Address.sol#155-160) is never used and should be removed
Address.functionStaticCall(addresses,bytes,bytes,bytes,bytes) (contracts/lib/Address.sol#162-167) is never used and should be removed
Address.getUin256Slot(bytes2) (contracts/traits/ERC1967Upgrade.sol#124-126) is never used and should be removed
ERC1967Upgrade.getAdmin() (contracts/traits/ERC1967Upgrade.sol#124-126) is never used and should be removed
ERC1967Upgrade.getBeacon() (contracts/traits/ERC1967Upgrade.sol#146-148) is never used and should be removed
ERC1967Upgrade.setBeacon(address) (contracts/traits/ERC1967Upgrade.sol#177) is never used and should be removed
ERC1967Upgrade.upgradeToAndCall(address,bytes,bytes) (contracts/traits/ERC1967Upgrade.sol#192-197) is never used and should be removed
ERC1967Upgrade.initializeERC1967Upgrade() (contracts/traits/ERC1967Upgrade.sol#198) is never used and should be removed
StorageSlot.getByte32Slot(bytes2) (contracts/lib/StorageSlot.sol#80-83) is never used and should be removed
StorageSlot.getUin256Slot(bytes2) (contracts/lib/StorageSlot.sol#84-87) is never used and should be removed
StorageSlot.getUin256Slot(bytes2) (contracts/lib/StorageSlot.sol#88-91) is never used and should be removed
StorageSlot.getUin256Slot(bytes2) (contracts/lib/StorageSlot.sol#92-95) is never used and should be removed
StorageSlot.getUin256Slot(bytes2) (contracts/lib/StorageSlot.sol#96-99) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

```

traits/Initializable.sol

No results from Slither.

traits/Marketplace.sol

```

ERC1967Upgrade, functionDelegateCall(address,bytes) (contracts/traitz/ERC1967Upgrade.sol#200-206) uses delegatecall to a input-controlled function id
  - (success, returnData) = target.delegatecall(data) (contracts/traitz/ERC1967Upgrade.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

AccessControl, grantRole(bytes32,address) (contractz/traits/AccessControl.sol#227-233) ignores return value by _roleMembers[role].add(account) (contracts/traitz/AccessControl.sol#230)
AccessControl, revokeRole(bytes32,address) (contractz/traits/AccessControl.sol#235-241) ignores return value by _roleMembers[role].remove(account) (contracts/traitz/AccessControl.sol#238)
ERC271NFT, _checkOnERC21Received(address,address,uint256,bytes) (contracts/traitz/ERC271NFT.sol#309->459) ignores return value by ERC271Received(magSender()),from,to,tokenId,data) (contracts/traitz/ERC271NFT.sol#440-455)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Variable _ERC21NFT, checkOnERC21Received(address,address,uint256,bytes), retval (contracts/traitz/ERC21NFT.sol#445) in ERC271NFT, checkOnERC21Received(address,address,uint256,bytes) (contracts/traitz/ERC21NFT.sol#438-455) potential y used before declaration, reason.length = 0 (contracts/traitz/ERC21NFT.sol#445)
Variable _ERC21NFT, checkOnERC21Received(address,address,uint256,bytes), result (contracts/traitz/ERC21NFT.sol#447) in ERC271NFT, checkOnERC21Received(address,address,uint256,bytes) (contracts/traitz/ERC21NFT.sol#438-455) potential y used before declaration, reason.length = 0 (contracts/traitz/ERC21NFT.sol#447)
Variable _ERC21NFT, checkOnERC21Received(address,address,uint256,bytes), reason (contracts/traitz/ERC21NFT.sol#448) in ERC271NFT, checkOnERC21Received(address,address,uint256,bytes) (contracts/traitz/ERC21NFT.sol#438-455) potential y used before declaration, reason.length = 0 (contracts/traitz/ERC21NFT.sol#448)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-use-of-local-variables

Reentrancy in ERC1967Upgrade,_upgradeToAndCallSecure(address,bytes,bool) (contracts/traitz/ERC1967Upgrade.sol#82-107):
  External calls:
    - _functionDelegateCall(newImplementation,data) (contracts/traitz/ERC1967Upgrade.sol#82)
    - _functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address,bytes)",newImplementation)) (contracts/traitz/ERC1967Upgrade.sol#82)
    - _functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address,bytes)",oldImplementation)) (contracts/traitz/ERC1967Upgrade.sol#100)
      - (success, returnData) = target.delegatecall(data) (contracts/traitz/ERC1967Upgrade.sol#100)
    - _owner.selector()
    - _upgradeTo(newImplementation) (contracts/traitz/ERC1967Upgrade.sol#105)
    - _upgradeTo(newImplementation) (contracts/traitz/ERC1967Upgrade.sol#105)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address, isContract(address) (contracts/lib/Address.sol#26-36) uses assembly
  - _isContract(address) (contracts/lib/Address.sol#31->36)
Address, verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#116->18)
  - INLINE ASM (contracts/lib/Address.sol#118-119)
EnumerableSet, values(EnumerableSet<T>) (contracts/lib/EnumerableSet.sol#273-282) uses assembly
  - _values(EnumerableSet<T>) (contracts/lib/EnumerableSet.sol#277-278)
EnumerableSet, values(EnumerableSet<T>Set) (contracts/lib/EnumerableSet.sol#356-355) uses assembly
  - INLINE ASM (contracts/lib/EnumerableSet.sol#350->353)
StorageSlot, getBooleanSlot(bytes32) (contracts/lib/StorageSlot.sol#51-55) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#50-54)
StorageSlot, getBooleanSlot(bytes32) (contracts/lib/StorageSlot.sol#60-64) uses assembly
  - INLINE ASM (contracts/lib/StorageSlot.sol#60-63)
StorageSlot, getBytes32Slot(bytes32) (contracts/lib/StorageSlot.sol#69-73)
  - INLINE ASM (contracts/lib/StorageSlot.sol#70->72)
StorageSlot, getBytes32Slot(bytes32) (contracts/lib/StorageSlot.sol#78-82)
  - INLINE ASM (contracts/lib/StorageSlot.sol#78-81)
Context, msg.sender() (contracts/traitz/Context.sol#16-15) uses assembly
ERC721NFT, checkOnERC721Received(address,address,uint256,bytes) (contracts/traitz/ERC721NFT.sol#438-455) uses assembly
  - INLINE ASM (contracts/traitz/ERC721NFT.sol#451)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-use

AccessControl, setupRole(bytes32,address) (contractz/traits/AccessControl.sol#422-424) is never used and should be removed
AccessControl, initializeAccessControl() (contractz/traits/AccessControl.sol#456-459) is never used and should be removed
Address, callWithValue(address,bytes,int) (contractz/lib/Address.sol#105->114) is never used and should be removed
Address, functionCallWithValue(address,bytes,int) (contractz/lib/Address.sol#105->114) is never used and should be removed
Address, functionCallStatic(address,bytes,int) (contractz/lib/Address.sol#111->143) is never used and should be removed
Address, functionStaticCall(address,bytes,int) (contractz/lib/Address.sol#111->160) is never used and should be removed
MassStorage, _setStorageSlot(bytes32,bytes32) (contractz/lib/MassStorage.sol#111->114) is never used and should be removed
Context, msg.data() (contracts/traitz/Context.sol#427-431) is never used and should be removed
Context, setTrustedForwarder(addresses) (contracts/traitz/Context.sol#10->10) is never used and should be removed
ERC165Interface, registerInterface(bytes) (contracts/traitz/ERC165Interface.sol#452-453) is never used and should be removed
ERC1822OPS, authorizeUpgrade(address) (contracts/traitz/ERC1822OPS.sol#160) is never used and should be removed
ERC1967Upgrade, initializeUpgrade(address) (contracts/traitz/ERC1967Upgrade.sol#160) is never used and should be removed
ERC1967Upgrade, _getMain() (contracts/traitz/ERC1967Upgrade.sol#124-126) is never used and should be removed
ERC1967Upgrade, _getStorageSlot(bytes32) (contracts/traitz/ERC1967Upgrade.sol#131-132) is never used and should be removed
ERC1967Upgrade, _setBeacon(address) (contracts/traitz/ERC1967Upgrade.sol#174) is never used and should be removed
ERC1967Upgrade, _upgradeDelegation(address,bytes,bytes,bytes) (contracts/traitz/ERC1967Upgrade.sol#193-197) is never used and should be removed
ERC1967Upgrade, _upgradeToAndCall(address,bytes,bytes,bytes) (contracts/traitz/ERC1967Upgrade.sol#204-208) is never used and should be removed
ERC1967Upgrade, _upgradeToAndCall(address,bytes,bytes,bytes) (contracts/traitz/ERC1967Upgrade.sol#204-208)
  - (success, returnData) = target.delegatecall(data) (contracts/traitz/ERC1967Upgrade.sol#208)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

ERC721NFT, _HTF_TRANSFER_TO_NON_RECEIVER (contracts/traitz/ERC721NFT.sol#80) is never used in ERC721NFT (contracts/traitz/ERC721NFT.sol#19-56)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

```

AUTOMATED TESTING

```

State Variables written after the call(s):
- listing.endTime = block.timestamp (contracts/traits/Marketplace.sol#324)
- listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#325)
- listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#326)
- listing.price = bidValueInListingCurrency (contracts/traits/Marketplace.sol#327)
- listing.duration = listing.endTime - listing.startTime (contracts/traits/Marketplace.sol#328)
- listing.bidAmount = bidValue (contracts/traits/Marketplace.sol#329)
listing.price = bidValueInListingCurrency (contracts/traits/Marketplace.sol#330)
listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#331)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#332)
listing.endTime = block.timestamp + listing.duration (contracts/traits/Marketplace.sol#333)
listing.bidCurrency = currency (contracts/traits/Marketplace.sol#334)
listing.startTime = block.timestamp - listing.duration (contracts/traits/Marketplace.sol#335)
listing.price = bidValueInListingCurrency (contracts/traits/Marketplace.sol#336)
listing.duration = listing.endTime - listing.startTime (contracts/traits/Marketplace.sol#337)
listing.bidAmount = bidValue (contracts/traits/Marketplace.sol#338)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#339)
listing.endTime = block.timestamp + listing.extensionDuration (contracts/traits/Marketplace.sol#340)
listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#341)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#342)
listing.endTime = block.timestamp + listing.extensionDuration (contracts/traits/Marketplace.sol#343)
listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#344)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#345)
listing.endTime = block.timestamp + listing.extensionDuration (contracts/traits/Marketplace.sol#346)
listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#347)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#348)
listing.endTime = block.timestamp + listing.extensionDuration (contracts/traits/Marketplace.sol#349)
listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#350)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#351)
listing.endTime = block.timestamp + listing.extensionDuration (contracts/traits/Marketplace.sol#352)
listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#353)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#354)
listing.endTime = block.timestamp + listing.extensionDuration (contracts/traits/Marketplace.sol#355)
listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#356)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#357)
listing.endTime = block.timestamp + listing.extensionDuration (contracts/traits/Marketplace.sol#358)
listing.buyerTokenWallet = address(buyerTokenWallet) (contracts/traits/Marketplace.sol#359)
listing.buyerPaymentWallet = address(magGender) (contracts/traits/Marketplace.sol#360)

Reference: https://github.com/crytic/slither/wlit/Detector-Documentation#reentrancy-vulnerabilities-1

MarketControl_isValidWallet(address) isValue (contracts/traits/Marketplace.sol#501) is a local variable never initialized
Reference: https://github.com/crytic/slither/wlit/Detector-Documentation#uninitialized-local-variables

AccessControl_grantRole(bytes32,address) (contracts/traits/AccessControl.sol#227-231) ignores return value by roleMembers[role].add(account) (contracts/traits/AccessControl.sol#230)
AccessControl_revokeRole(bytes32,address) (contracts/traits/AccessControl.sol#235-241) ignores return value by roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#238)
Marketplace_isValidWallet(address) (contracts/traits/Marketplace.sol#500-506) ignores return value by config.isValidWallet(wallet) (contracts/traits/Marketplace.sol#501-505)
Reference: https://github.com/crytic/slither/wlit/Detector-Documentation#uninitialized-local-variables

Variable Marketplace_isValidWallet(address) isValue (contracts/traits/Marketplace.sol#501) in Marketplace_isValidWallet(address) (contracts/traits/Marketplace.sol#500-506) potentially used before declaration: isValue (contracts/traits/Marketplace.sol#501)
Reference: https://github.com/crytic/slither/wlit/Detector-Documentation#pre-declaration-use-of-local-variables

Reentrancy in Marketplace._createListing(uint256,uint256,string,address,address,uint8) (contracts/traits/Marketplace.sol#442-495):
External calls:
- require(bool,string) (_isValidWallet(magGender),ERR UNSUPPORTED TOKEN WALLET) (contracts/traits/Marketplace.sol#459)
- _config.iisValidWallet(wallet) (contracts/traits/Marketplace.sol#501-505)
- _transferToken(trustedToken.transferFrom(from,to,tokenId)) (contracts/traits/Marketplace.sol#460)
- _ERC721(trustedToken.transferFrom(from,to,tokenId)) (contracts/traits/Marketplace.sol#413-515)
State variables written after the call(s):
- _gasLimitConfigured(_gasLimitLow,_gasLimitHigh) (contracts/traits/Marketplace.sol#461-466)
Reentrancy in Marketplace._reconfigureConfiguration() (contracts/traits/Marketplace.sol#425-439):
External calls:
- _gasLimitConfigured(_gasLimitLow,_gasLimitHigh) (contracts/traits/Marketplace.sol#426)
State variables written after the call(s):
- _gasLimitConfigured(_gasLimitLow,_gasLimitHigh) (contracts/traits/Marketplace.sol#427)
Reentrancy in Marketplace._reconfigureConfiguration() (contracts/traits/Marketplace.sol#426-439):
External calls:
- (success) = user.call(gas:_gasLimit,value:_amount) () (contracts/traits/Marketplace.sol#427)
- _pendingWithdrawals[_user] = pendingWithdrawals[_user] + amount (contracts/traits/Marketplace.sol#432)
Reentrancy in Marketplace.placeBid(uint256,address,uint256) (contracts/traits/Marketplace.sol#379-377):
External calls:
- require(bool,string) (_isValidWallet(buyerTokenWallet),ERR UNSUPPORTED TOKEN WALLET) (contracts/traits/Marketplace.sol#380)
- _config.iisValidWallet(wallet) (contracts/traits/Marketplace.sol#501-505)
- _finalization(_gasLimitLow,_gasLimitHigh) (contracts/traits/Marketplace.sol#381)
- _ERC721(trustedToken.transferFrom(from,to,tokenId)) (contracts/traits/Marketplace.sol#413-515)
- (success) = user.call(gas:_gasLimit,value:_amount) () (contracts/traits/Marketplace.sol#462)
- _ERC20(_contractAddress).transferFrom(magGender,treasury,platformFeeableAmount) (contracts/traits/Marketplace.sol#611)
- _ERC20(_contractAddress).transferFrom(magGender,seller,payableAmount) (contracts/traits/Marketplace.sol#612)
External calls sending eth:
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#382)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#383)
State variables written after the call(s):
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#382)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#383)
Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#550-563):
External calls:
- _reducedGasLimit(_gasLimitLow,_gasLimitHigh) (contracts/traits/Marketplace.sol#560)
- (success) = user.call(gas:_gasLimit,value:_amount) () (contracts/traits/Marketplace.sol#560)
State variables written after the call(s):
- hasConfiguration = true (contracts/traits/Marketplace.sol#577)
Reference: https://github.com/crytic/slither/wlit/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#550-563):
External calls:
- _ERC721(trustedToken).transferFrom(address(this),listing.sellerTokenWallet,tokenId) (contracts/traits/Marketplace.sol#560)
Event emitted after the call(s):
- _finaliseListing(listingId,uint256,uint256,uint256,uint256,uint256) = _cancelGasLimits() (contracts/traits/Marketplace.sol#560)
Reentrancy in Marketplace._createListing(uint256,uint256,string,address,address,uint8) (contracts/traits/Marketplace.sol#442-495):
External calls:
- require(bool,string) (_isValidWallet(magGender),ERR UNSUPPORTED TOKEN WALLET) (contracts/traits/Marketplace.sol#459)
- _config.iisValidWallet(wallet) (contracts/traits/Marketplace.sol#501-505)
- _transferToken(magGender).transferFrom(this,to,tokenId) (contracts/traits/Marketplace.sol#460)
- _ERC20(_contractAddress).transferFrom(from,to,tokenId) (contracts/traits/Marketplace.sol#413-515)
Event emitted after the call(s):
- ListingCreated(listingType,listingId,magGender,sellerTokenWallet,buyerPaymentWallet,listing.price,listing.duration,listing.extensionDuration,listing.endTime) (contracts/traits/Marketplace.sol#485-495)
7) Reentrancy in Marketplace._finaliseListing(uint256) (contracts/traits/Marketplace.sol#564-594):
External calls:
- _transferFunds(_listingSellerPayableAmount,_listingBuyerPayableAmount) (contracts/traits/Marketplace.sol#576)
- (platformFee,sellerPayableAmount) = _distributeFunds(listingSellerPayableAmount,listing.buyerPayableAmount) (contracts/traits/Marketplace.sol#577-581)
- (success) = user.call(gas:_gasLimit,value:_amount) () (contracts/traits/Marketplace.sol#587)
- _ERC20(_contractAddress).transferFrom(magGender,seller,payableAmount) (contracts/traits/Marketplace.sol#612)
External calls sending eth:
- (platformFee,sellerPayableAmount) = _distributeFunds(listingSellerPayableAmount,listing.buyerPayableAmount) (contracts/traits/Marketplace.sol#577-581)
- (success) = user.call(gas:_gasLimit,value:_amount) () (contracts/traits/Marketplace.sol#587)
Event emitted after the call(s):
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#577)
Reentrancy in Marketplace._finaliseListing(uint256) (contracts/traits/Marketplace.sol#564-594):
External calls:
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#577)
- require(bool,string) (_isValidWallet(magGender),ERR UNSUPPORTED TOKEN WALLET) (contracts/traits/Marketplace.sol#580)
- _config.iisValidWallet(wallet) (contracts/traits/Marketplace.sol#501-505)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#577)
- _ERC721(trustedToken).transferFrom(from,to,tokenId) (contracts/traits/Marketplace.sol#413-515)
- _ERC20(_contractAddress).transferFrom(magGender,treasury,platformFeeableAmount) (contracts/traits/Marketplace.sol#611)
- _ERC20(_contractAddress).transferFrom(magGender,seller,payableAmount) (contracts/traits/Marketplace.sol#612)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#577)
Event emitted after the call(s):
- ListingFinalized(listingType,listingId,magGender,sellerTokenWallet,buyerPaymentWallet,listing.price,listing.duration,listing.extensionDuration,listing.endTime,listing.platformFee,sellerPayableAmount) (contracts/traits/Marketplace.sol#585-593)
Reentrancy in ERC1967Upgrade.upgradeToAndCall(address,bytes,func) (contracts/traits/ERC1967Upgrade.sol#107):
External calls:
- _functionDelegateCall(_newImplementation,data) (contracts/traits/ERC1967Upgrade.sol#92)
- (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#208)
- _functionDelegateCall(_newImplementation,abi.encodeWithSignature(_upgradeTo(address),oldImplementation)) (contracts/traits/ERC1967Upgrade.sol#100)
- (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
Event emitted after the call(s):
- _Upgraded(newImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
- _Upgraded(oldImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#299-377):
External calls:
- require(bool,string) (_isValidWallet(magGender),ERR UNSUPPORTED TOKEN WALLET) (contracts/traits/Marketplace.sol#306)
- _config.iisValidWallet(wallet) (contracts/traits/Marketplace.sol#501-505)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#325)
- _ERC721(trustedToken).transferFrom(from,to,tokenId) (contracts/traits/Marketplace.sol#413-515)
- _ERC20(_contractAddress).transferFrom(magGender,treasury,platformFeeableAmount) (contracts/traits/Marketplace.sol#611)
- _ERC20(_contractAddress).transferFrom(magGender,seller,payableAmount) (contracts/traits/Marketplace.sol#612)
- _finaliseListing(listingId) (contracts/traits/Marketplace.sol#325)
Event emitted after the call(s):
- ListingCancelled(listingType,listingId,magGender,sellerTokenWallet,buyerPaymentWallet,listing.price,listing.duration,listing.extensionDuration,listing.endTime) (contracts/traits/Marketplace.sol#377)
Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#299-377):
External calls:
- require(bool,string) (_isValidWallet(magGender),ERR UNSUPPORTED TOKEN WALLET) (contracts/traits/Marketplace.sol#306)
- _config.iisValidWallet(wallet) (contracts/traits/Marketplace.sol#501-505)
Event emitted after the call(s):
- AuctionCancelled(listingId,magGender,bidValue,listing.endTime) (contracts/traits/Marketplace.sol#344)
Reentrancy in Marketplace._cancelListing(uint256) (contracts/traits/Marketplace.sol#299-377):
External calls:
- require(bool,string) (_isValidWallet(magGender),ERR UNSUPPORTED TOKEN WALLET) (contracts/traits/Marketplace.sol#306)
- _config.iisValidWallet(wallet) (contracts/traits/Marketplace.sol#501-505)
Event emitted after the call(s):
- AuctionCancelled(listingId,magGender,bidValue,listing.endTime) (contracts/traits/Marketplace.sol#344)
Reentrancy in Marketplace.withdrawFor(address) (contracts/traits/Marketplace.sol#401-407):
External calls:
- _sendFunds(amount,_gasLimitHigh) (contracts/traits/Marketplace.sol#405)
- (success) = user.call(gas:_gasLimit,value:_amount) () (contracts/traits/Marketplace.sol#427)
Event emitted after the call(s):
- Withdrawal(user,amount) (contracts/traits/Marketplace.sol#406)
Reference: https://github.com/crytic/slither/wlit/Detector-Documentation#reentrancy-vulnerabilities-3

```

traits/MarketplaceConfig.sol

```
ERC1967Upgrade, function.delegatecall(address, bytes) (contracts/traits/ERC1967Upgrade.sol#000-206) uses delegatecall to a input-controlled function id
  - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#020)
Reference: https://github.com/crytic/solcifer-wiki/Detector-Documents#function-controlled-delegatecall

MarketplaceConfig, existsUserWallet(address)_isValid (contracts/traits/MarketplaceConfig.sol#45) is a local variable never initialized
Reference: https://github.com/crytic/solcifer-wiki/Detector-Documents#function-uninitialized-local-variables

AccessControl_grantRole(bytes2, address) (contracts/traits/AccessControl.sol#227-233) ignores return value by _roleMembers[role].add(account) (contracts/traits/AccessControl.sol#230)
  - (success,returnData) = _roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#238)
MarketplaceConfig, existsUserWallet(address)_isValid[144-150] ignores return value by _existsUserWallet(wallet) (contracts/traits/MarketplaceConfig.sol#145-149)
Reference: https://github.com/crytic/solcifer-wiki/Detector-Documents#unhandled-return

Variable MarketplaceConfig.existsUserWallet(address)_isValid[150] (contracts/traits/MarketplaceConfig.sol#145+) in MarketplaceConfig.existsUserWallet(address) (contracts/traits/MarketplaceConfig.sol#144-150) potentially used before declaration
  - (success,returnData) = _existsUserWallet(wallet) (contracts/traits/MarketplaceConfig.sol#144)
Reference: https://github.com/crytic/solcifer-wiki/Detector-Documents#pre-declaration-of-local-variables

Reentrancy in ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#82-107):
  - _function.delegatecall(newImplementation,data) (contracts/traits/ERC1967Upgrade.sol#45)
    - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
    - _functionDelegateCallWithHashSig(sha3(implementation._upgradeTo(address),oldImplementation)) (contracts/traits/ERC1967Upgrade.sol#100)
      - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
Event emitted after _call1(data):
  - _upgradeTo(newImplementation) (contracts/traits/ERC1967Upgrade.sol#55)
    - _upgradeTo(newImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
Reference: https://github.com/crytic/solcifer-wiki/Detector-Documents#function-reentrancy-vulnerabilities-3

Address.getAddress() (contracts/lib/Address.sol#2-36) uses assembly
  - INLINE ASM (contract=Address.sol#180-183)
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#165-188) uses assembly
  - INLINE ASM (contract=Address.sol#180-183)
EnumerableSet.value(uint) (contracts/lib/EnumerableSet.sol#277-279) uses assembly
  - INLINE ASM (contract=EnumerableSet.sol#277-279)
EnumerableSet.value(uint) (contracts/lib/EnumerableSet.sol#346-355) uses assembly
EnumerableSet.value(uint) (contracts/lib/EnumerableSet.sol#346-355) uses assembly

StorageSlot.getAddressSlot(address) (contracts/lib/StorageSlot.sol#1-55) uses assembly
  - INLINE ASM (contract=StorageSlot.sol#52-54)
StorageSlot.getAddressSlot(address) (contracts/lib/StorageSlot.sol#60-64) uses assembly
  - INLINE ASM (contract=StorageSlot.sol#60-64)
StorageSlot.getByte32Slot(bytes32) (contracts/lib/StorageSlot.sol#70-73) uses assembly
  - INLINE ASM (contract=StorageSlot.sol#70-73)
StorageSlot.getByte32Slot(bytes32) (contracts/lib/StorageSlot.sol#175-182) uses assembly
  - INLINE ASM (contract=StorageSlot.sol#70-73)
Context.msgSender() (contracts/traits/Context.sol#4-25) uses assembly
  - (success,returnData) = msg.sender (contracts/traits/Context.sol#4)

Reference: https://github.com/crytic/solcifer-wiki/Detector-Documents#assembly-use
```

AUTOMATED TESTING

traits/Pausable.sol

```
Context.msgGender() (contracts/traits/Context.sol#16-25) uses assembly
- INLINE ASM (contracts/traits/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wl/better-Docmentation#assembly-use

Context.isTrustedForwarder(address) (contracts/traits/Context.sol#18-19) is never used and should be removed
Context.setMsgSender() (contracts/traits/Context.sol#14-15) is never used and should be removed
Context.setTrustedForwarder(address) (contracts/traits/Context.sol#18-19) is never used and should be removed
Pausable.pause() (contracts/traits/Pausable.sol#14-15) is never used and should be removed
Pausable.initializePausable() (contracts/traits/Pausable.sol#30-32) is never used and should be removed
Reference: https://github.com/crytic/slither/wl/better-Docmentation#dead-code
```

traits/MetadataDB.sol

```
ERC1967Upgrade..functionDelegateCall(address,bytes) (contracts/traits/ERC1967Upgrade.sol#200-206) uses delegatecall to a input-controlled function id
Reference: https://github.com/crytic/slither/wl/better-Docmentation#controlled-delegatecall

AccessControl.grantRole(bytes32,address) (contracts/traits/AccessControl.sol#27-23) ignores return value by _roleMembers[role].add(account) (contracts/traits/AccessControl.sol#20)
AccessControl.revokeRole(bytes32,address) (contracts/traits/AccessControl.sol#23-24) ignores return value by _roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#28)
MetadataDB.addAttribute(string,string,MetadataDB.AttributeType,bool) (contracts/traits/MetadataDB.sol#172-193) ignores return value by _attributeName.add(attribute) (contracts/traits/MetadataDB.sol#151)
MetadataDB.setBooleanValue(uint256,uint256) (contracts/traits/MetadataDB.sol#173-194) ignores return value by _tokenIds.add(tokenId) (contracts/traits/MetadataDB.sol#154)
MetadataDB.setBooleanValue(uint256,uint256,bool) (contracts/traits/MetadataDB.sol#240-249) ignores return value by _tokenIds.add(tokenId) (contracts/traits/MetadataDB.sol#236)
MetadataDB.setBooleanValue(uint256,uint256,bool) (contracts/traits/MetadataDB.sol#242-259) ignores return value by _tokenIds.add(tokenId) (contracts/traits/MetadataDB.sol#255)
Reference: https://github.com/crytic/slither/wl/better-Docmentation#return-value

Reentrancy in ERC1967Upgrade..upgradeToAndCallSuccess(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#2-107):
- _functionDelegateCall(implementation,data) (contracts/traits/ERC1967Upgrade.sol#92)
  - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
- _functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address,bytes)",implementation)) (contracts/traits/ERC1967Upgrade.sol#100)
  - target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
Event emitted after this call(s):
- _Upgraded(newImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
Reference: https://github.com/crytic/slither/wl/better-Docmentation#reentrancy-vulnerabilities

Address.isContract(address) (contracts/lib/Address.sol#2-36) uses assembly
- INLINE ASM (contracts/lib/Address.sol#32-34)
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#168-188) uses assembly
- INLINE ASM (contracts/lib/Address.sol#170-183)
EnumerableSet.contains(EnumerableSet.AddressSet, address) (contracts/lib/EnumerableSet.sol#273-282) uses assembly
- INLINE ASM (contracts/lib/EnumerableSet.sol#277-279)
EnumerableSet.value(EnumerableSet.UintSet, uint) (contracts/lib/EnumerableSet.sol#194-35) uses assembly
StorageSlot.getAddressSlot(bytes32) (contracts/lib/StorageSlot.sol#51-58)
  - INLINE ASM (contracts/lib/StorageSlot.sol#52-54)
StorageSlot.getBoolSlot(bytes32) (contracts/lib/StorageSlot.sol#140-64)
  - INLINE ASM (contracts/lib/StorageSlot.sol#64-63)
StorageSlot.getByteSlot(bytes32) (contracts/lib/StorageSlot.sol#69-73) uses assembly
StorageSlot.getU16Slot(bytes32) (contracts/lib/StorageSlot.sol#74-78)
  - INLINE ASM (contracts/lib/StorageSlot.sol#77-81)
Context._msgSender() (contracts/Context.sol#19-21) uses assembly
- INLINE ASM (contracts/Context.sol#19-21)
Reference: https://github.com/crytic/slither/wl/better-Docmentation#assembly-use

AccessControl.setupRole(bytes32,address) (contracts/traits/AccessControl.sol#212-214) is never used and should be removed
AccessControl.initializeAccessControl() (contracts/traits/AccessControl.sol#56-59) is never used and should be removed
Address.functionCall(address,bytes) (contracts/lib/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/lib/Address.sol#105-114) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,bytes) (contracts/lib/Address.sol#121-133) is never used and should be removed
Address.functionStaticCall(addresses,bytes,string) (contracts/lib/Address.sol#145-160) is never used and should be removed
Address.sendValue(address,uint256) (contracts/lib/Address.sol#164-165) is never used and should be removed
Context.setTrustedForwarder(address) (contracts/traits/Context.sol#18-19) is never used and should be removed
ERC1967Interface.initializeERC1967Interface() (contracts/traits/ERC1967Interface.sol#12-15) is never used and should be removed
ERC1967Upgrade.authorizeUpgrade(address) (contracts/traits/ERC1967Upgrade.sol#182-183) is never used and should be removed
ERC1967Upgrade.initializeERC1967Upgrade() (contracts/traits/ERC1967Upgrade.sol#184-185) is never used and should be removed
ERC1967Upgrade.upgrade(address,bytes) (contracts/traits/ERC1967Upgrade.sol#192-193) is never used and should be removed
ERC1967Upgrade.upgradeTo(address,bytes) (contracts/traits/ERC1967Upgrade.sol#194-195) is never used and should be removed
EnumerableSet.set(EnumerableSet.Set,bytes32) (contracts/lib/EnumerableSet.sol#161-173) is never used and should be removed
EnumerableSet.set(EnumerableSet.Set,bytes32,bytes) (contracts/lib/EnumerableSet.sol#161-173) is never used and should be removed
EnumerableSet.set(EnumerableSet.Set,bytes32,bytes,bytes) (contracts/lib/EnumerableSet.sol#161-173) is never used and should be removed
EnumerableSet.set(EnumerableSet.Set,bytes32,bytes,bytes,bytes) (contracts/lib/EnumerableSet.sol#161-173) is never used and should be removed
EnumerableSet.set(EnumerableSet.AddressSet,address) (contracts/lib/EnumerableSet.sol#207-209) is never used and should be removed
EnumerableSet.set(EnumerableSet.AddressSet,address,bytes) (contracts/lib/EnumerableSet.sol#207-209) is never used and should be removed
EnumerableStringSet.remove(EnumerableStringSet.StringSet,bytes) (contracts/lib/EnumerableStringSet.sol#127-129) is never used and should be removed
EnumerableStringSet.value(EnumerableStringSet.StringSet,bytes) (contracts/lib/EnumerableStringSet.sol#127-129) is never used and should be removed
EnumerableStringSet.value(EnumerableStringSet.StringSet,bytes,bytes) (contracts/lib/EnumerableStringSet.sol#127-129) is never used and should be removed
EnumerableStringSet.value(EnumerableStringSet.StringSet,bytes,bytes,bytes) (contracts/lib/EnumerableStringSet.sol#127-129) is never used and should be removed
StorageSlot.getAddressSlot(bytes32) (contracts/lib/StorageSlot.sol#67-73) is never used and should be removed
StorageSlot.getBoolSlot(bytes32) (contracts/lib/StorageSlot.sol#78-82) is never used and should be removed
StorageSlot.getByteSlot(bytes32) (contracts/lib/StorageSlot.sol#83-87) is never used and should be removed
String.toHex(string,uint256) (contracts/lib/Strings.sol#19-50) is never used and should be removed
String.toInt(string,uint256) (contracts/lib/Strings.sol#14-34) is never used and should be removed
Upgradable.initializeUpgradable(address) (contracts/traits/Upgradable.sol#16-24) is never used and should be removed
Reference: https://github.com/crytic/slither/wl/better-Docmentation#dead-code
```

AUTOMATED TESTING

traits/RarioNFT.sol

```

ERC1967Upgrade, function.delegatecall(address,bytes) (contracts/traits/ERC1967Upgrade.sol#0x10-0x20) uses delegatecall to a input-controlled function id
  - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#0x20)
Reference: https://github.com/crytic/slither/wikidetectorDocumentationcontrolled-delegatecall

AccessControl_grantRole(bytes32,address) (contracts/traits/AccessControl.sol#227-233) ignores return value by roleMembers[role].add(account) (contracts/traits/AccessControl.sol#0x20)
AccessControl_revokeRole(bytes32,address) (contracts/traits/AccessControl.sol#0x235-0x240) ignores return value by roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#0x20)
ERC721NFT_checkOnERC721Received(address,address,uint256,bytes) (contracts/traits/ERC721NFT.sol#0x438-0x459) ignores return value by IERC721Receiver(to).onERC721Received(msg.sender(),from,to tokenId, data) (contracts/traits/ERC721NFT.sol#0x445-0x455)
Reference: https://github.com/crytic/slither/wikidetectorDocumentationunused-return

RareNFT_getAttribute(bytes16,int32), name (contracts/traits/RareNFT.sol#0x10-0x16) shadows
  - IERC721Metadata.name (contracts/traits/IERC721Metadata.sol#0x10) (function)
RareNFT_getAttribute(string), name (contracts/traits/RareNFT.sol#0x10-0x16) shadows
  - IERC721Metadata.name (contracts/traits/IERC721Metadata.sol#0x10) (function)
  - IERC721Metadata.name (contracts/traits/IERC721Metadata.sol#0x10) (function)
Reference: https://github.com/crytic/slither/wikidetectorDocumentationlocal-variable-shadowing

Variable "ERC721NFT__checkOnERC721Received(address,address,uint256,bytes).retval" (contracts/traits/ERC721NFT.sol#0x448-0x459) potential y use before declaration, retval == IERC721Receiver.onERC721Received.selector (contracts/traits/ERC721NFT.sol#0x446)
Variable "ERC721NFT__checkOnERC721Received(address,address,uint256,bytes).reason" (contracts/traits/ERC721NFT.sol#0x447) in ERC721NFT__checkOnERC721Received(address,address,uint256,bytes) (contracts/traits/ERC721NFT.sol#0x438-0x456) potential y use before declaration, reason == uint256(32) or reason == uint256(32) * reason, load(uint256(32)) (contracts/traits/ERC721NFT.sol#0x445)
Variable "ERC721NFT__checkOnERC721Received(address,address,uint256,bytes).season" (contracts/traits/ERC721NFT.sol#0x446) in ERC721NFT__checkOnERC721Received(address,address,uint256,bytes) (contracts/traits/ERC721NFT.sol#0x438-0x455) potential y use before declaration, revert(uint256,uint256)(32 or reason, load(uint256(32)) (contracts/traits/ERC721NFT.sol#0x445)
Reference: https://github.com/crytic/slither/wikidetectorDocumentationpre-declaration-use-of-local-variables

Reentrancy in ERC1967Upgrade_upgradeToAndCall(caller, address, bytes, bool) (contracts/ERC1967Upgrade.sol#0x82-0x107):
  External calls:
    - _functionImplementation((newImplementation), (data)) (contracts/traits/ERC1967Upgrade.sol#0x9)
      - (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#0x20)
        - _functionImplementation((newImplementation), (bytes32(gasPrice,upgradeTo(address),oldImplementation))) (contracts/traits/ERC1967Upgrade.sol#0x100)
        Event emitted after the call:
    - Upgraded(newImplementation) (contracts/traits/ERC1967Upgrade.sol#0x105)
      upgradeTo((newImplementation)) (contracts/traits/ERC1967Upgrade.sol#0x105)
Reference: https://github.com/crytic/slither/wikidetectorDocumentationreentrancy-and-call-depth
```

traits/RarioEthereumNFT.sol

[traits/RarioPolygonNFT.sol](#)
ERC1967Upgrade:delegatecall,delegatedCall,upgrade
+ success,returnData = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204) uses delegatecall to a input-controlled function id
Reference: <https://github.com/crytic/sleifer/wiki/Detector-Documentation#controlled-delegatecall>

AccessControl_grantRoleByByte33:address (contracts/traits/AccessControl.sol#237-238) ignores return value by _roleMembers[role].add(account) (contracts/traits/AccessControl.sol#230)
AccessControl_revokeRoleByByte33:address (contracts/traits/AccessControl.sol#235-236) ignores return value by _roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#238)
ERC721NFT_checkOnERC721Received(address,address,uint256,bytes) (contracts/traits/ERC721NFT.sol#438-459) ignores return value by IERC721Receiver.onERC721Received(msgSender(),from,to,tokenId_,data) (contracts/traits/ERC721NFT.sol#440-455)
Reference: <https://github.com/crytic/sleifer/wiki/Detector-Documentation#unused-return>

RarioNFT_getAttributeByIndex:uint256 (contracts/traits/RarioNFT.sol#466) shadows:
- IERC721Metadata.name() (contracts/traits/ERC721Metadata.sol#82-94) (function)
- IERC721Metadata.name() (contracts/interfaces/IERC721Metadata.sol#15) (function)
RarioNFT_getAttributeByIndex:uint256 (contracts/traits/RarioNFT.sol#466) shadows:
- IERC721Metadata.name() (contracts/traits/ERC721Metadata.sol#82-94) (function)
- IERC721Metadata.name() (contracts/interfaces/IERC721Metadata.sol#15) (function)
Reference: <https://github.com/crytic/sleifer/wiki/Detector-Documentation#local-variable-shadowing>

Possible Variable IERC721NFT_checkOnERC721Received(address,address,uint256,bytes).recv (contracts/traits/ERC721NFT.sol#445) in ERC721NFT_checkOnERC721Received(address,address,uint256,bytes) (contracts/traits/ERC721NFT.sol#438-459) potentialy used before declaration; reason: length == 0 (contracts/traits/ERC721NFT.sol#445)
Variable IERC721NFT_checkOnERC721Received(address,address,uint256,bytes).recv (contracts/traits/ERC721NFT.checkOnERC721Received(address,address,uint256,bytes) (contracts/traits/ERC721NFT.sol#438-459) potentialy used before declaration; reason: length == 0 (contracts/traits/ERC721NFT.sol#445)
Variable IERC721NFT_checkOnERC721Received(address,address,uint256,bytes).recv (contracts/traits/ERC721NFT.checkOnERC721Received(address,address,uint256,bytes) (contracts/traits/ERC721NFT.sol#438-459) potentialy used before declaration; reason: length == 0 (contracts/traits/ERC721NFT.sol#445)
Reference: <https://github.com/crytic/sleifer/wiki/Detector-Documentation#pre-declaration-use-of-local-variables>

Renentrancy in ERC1967Upgrade_upgradeAndCallSecure(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#107):
External calls:
- _functionImplementationNewImplementation(data) (contracts/traits/ERC1967Upgrade.sol#107)
- (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
- _functionImplementationCallNewImplementation(abi.encodeWithSignature(upgradeTo,address),oldImplementation) (contracts/traits/ERC1967Upgrade.sol#100)
Event emitted after the call(s):
- upgrader(target.delegatecall(data)) (events/ERC1967Upgrade.sol#15)
- upgraded(newImplementation) (contracts/traits/ERC1967Upgrade.sol#158)
- upgradedToNewImplementation (contracts/traits/ERC1967Upgrade.sol#105)
Reference: <https://github.com/crytic/sleifer/wiki/Detector-Documentation#renentrancy-vulnerabilities-3>

Address,_registerAddress (contracts/lib/Address.sol#26-36) uses assembly
- INLNE ASM (contracts/lib/Address.sol#26-36) uses assembly
Addresses.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#169-180) uses assembly
- INLNE ASM (contracts/lib/Address.sol#169-180)
EnumerableSet.UintSet (contracts/lib/EnumerableSet.sol#277-299) uses assembly
- INLNE ASM (contracts/lib/EnumerableSet.sol#277-299)
EnumerableSet.values(EnumerableSet.UintSet) (contracts/lib/EnumerableSet.sol#346-355) uses assembly
- INLNE ASM (contracts/lib/EnumerableSet.sol#346-355)
StorageSlot.getAddressSlot(bytes2) (contracts/lib/StorageSlot.sol#451-55) uses assembly
- INLNE ASM (contracts/lib/StorageSlot.sol#452-54)
StorageSlot.getByte32Slot(bytes2) (contracts/lib/StorageSlot.sol#460-53) uses assembly
- INLNE ASM (contracts/lib/StorageSlot.sol#461-53)
StorageSlot.getBytes32Slot(bytes2) (contracts/lib/StorageSlot.sol#469-73) uses assembly
- INLNE ASM (contracts/lib/StorageSlot.sol#470-73)
StorageSlot.getUint256Slot(bytes2) (contracts/lib/StorageSlot.sol#478-82) uses assembly
- INLNE ASM (contracts/lib/StorageSlot.sol#479-82)
Context._msgSender() (contracts/context/Context.sol#14-21) uses assembly
- INLNE ASM (contracts/context/Context.sol#14-21)
ERC721NFT_checkOnERC721Received(address,address,uint256,bytes) (contracts/traits/ERC721NFT.sol#438-459) uses assembly
- INLNE ASM (contracts/traits/ERC721NFT.sol#438-459)
Reference: <https://github.com/crytic/sleifer/wiki/Detector-Documentation#assembly>

traits/ReentrancyGuard.sol

```

ReentrancyGuard.initializeReentrancyGuard() (contracts/traits/ReentrancyGuard.sol#39-41) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#read-code

ReentrancyGuard._gap (contracts/traits/ReentrancyGuard.sol#63) is never used in ReentrancyGuard (contracts/traits/ReentrancyGuard.sol#22-64)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

traits/Upgradeable.sol

```

```

AccessControl.setupRole(bytes32,address) (contracts/traits/AccessControl.sol#212-214) is never used and should be removed
AccessControl.initializeAccessControl() (contracts/traits/AccessControl.sol#56-59) is never used and should be removed
Address.functionCall(address,bytes,Uint256) (contracts/lib/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/lib/Address.sol#108-114) is never used and should be removed
Address.functionCallWithReturnData(address,bytes,Uint256) (contracts/lib/Address.sol#122-133) is never used and should be removed
Address.functionStaticCall(address,bytes,Uint256) (contracts/lib/Address.sol#141-143) is never used and should be removed
Address.sendValue(address,Uint256) (contracts/lib/Address.sol#154-160) is never used and should be removed
Context.setTrustedForwarder(address) (contracts/lib/Context.sol#10-11) is never used and should be removed
Context.setTrustedForwarderForwards(address) (contracts/lib/Context.sol#11-12) is never used and should be removed
ERC165Interface.initializeERC165Interface() (contracts/traits/ERC165Interface.sol#20-25) is never used and should be removed
ERC165Interface.isInterfaceSupported(bytes4) (contracts/traits/ERC165Interface.sol#26-31) is never used and should be removed
ERC1822UDUFS.initializeERC1822UDUFS() (contracts/traits/ERC1822UDUFS.sol#2-3) is never used and should be removed
ERC196Upgrade._changeAdmin(address) (contracts/traits/ERC196Upgrade.sol#141-144) is never used and should be removed
ERC196Upgrade._setAdmin(address) (contracts/traits/ERC196Upgrade.sol#145-148) is never used and should be removed
ERC196Upgrade._upgradeAndCall(address,bytes,bool) (contracts/traits/ERC196Upgrade.sol#149-152) is never used and should be removed
ERC196Upgrade._upgradeAndCallWithReturnData(address,bytes,Uint256) (contracts/traits/ERC196Upgrade.sol#153-156) is never used and should be removed
ERC721INFT._initializeERC721INFT(string,string) (contracts/traits/ERC721INFT.sol#7-13) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#141-143) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#144-146) is never used and should be removed
EnumerableSet.length(EnumerableSet_Length) (contracts/lib/EnumerableSet.sol#147-149) is never used and should be removed
EnumerableSet.remove(EnumerableSet_Remove) (contracts/lib/EnumerableSet.sol#149-151) is never used and should be removed
EnumerableSet.value(EnumerableSet_Value) (contracts/lib/EnumerableSet.sol#152-154) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#155-157) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#158-160) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#161-163) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#164-166) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#167-169) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#170-172) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#173-175) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#176-178) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#179-181) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#182-184) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#185-187) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#188-190) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#191-193) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#194-196) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#197-199) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#200-202) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#203-205) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#206-208) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#209-211) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#212-214) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#215-217) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#218-220) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#221-223) is never used and should be removed
EnumerableSet.contains(EnumerableSet_Contains) (contracts/lib/EnumerableSet.sol#224-226) is never used and should be removed
EnumerableSet.set(EnumerableSet_Set) (contracts/lib/EnumerableSet.sol#227-229) ignores return value by _roleMembers[role].add(account) (contracts/traits/AccessControl.sol#230)
AccessControl._removeRole(bytes32,address) (contracts/traits/AccessControl.sol#230-240) ignores return value by _roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#230)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions

ERC721NFT._TRANSFER_TO_NON_RECEIVER (contracts/traits/ERC721NFT.sol#140) is never used in RatioPolygonNFT (contracts/traits/RatioPolygonNFT.sol#8-126)
Reference: https://github.com/Crytic/slither/wiki/Detector-Documentation#unused-state-variable

RatioPolygonNFT (contracts/crypt/RatioPolygonNFT.sol#1-26) does not implement functions:
- ERC1822UDUFS._authorizer(address) (contracts/traits/ERC1822UDUFS.sol#40)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions

ERC721NFT._TRANSFER_TO_NON_RECEIVER (contracts/traits/ERC721NFT.sol#140) is never used in RatioPolygonNFT (contracts/traits/RatioPolygonNFT.sol#8-126)
Reference: https://github.com/Crytic/slither/wiki/Detector-Documentation#unused-state-variable

```

```

traits/UserDB.sol
ERC1967Upgrade.functionDelegateCall(address,bytes) (contracts/traits/AccessControl.sol#200-206) uses delegatecall to a input-controlled function id
- (success,returnData) = target.delegatecall(data) (contracts/traits/ERC1967Upgrade.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#controlled-delegatecall

AccessControl.grantRole(bytes32,address) (contracts/traits/AccessControl.sol#227-233) ignores return value by _roleMembers[role].add(account) (contracts/traits/AccessControl.sol#230)
AccessControl.revokeRole(bytes32,address) (contracts/traits/AccessControl.sol#234-239) ignores return value by _roleMembers[role].remove(account) (contracts/traits/AccessControl.sol#238)
UserDB.addUser(bytes32,address) (contracts/traits/UserDB.sol#1-11) ignores return value by raxidc.add(raxid) (contracts/traits/UserDB.sol#47)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Reentrancy in ERC1967Upgrade._upgradeToAndCall(ercup, address, bytes, bool) (contracts/traits/ERC1967Upgrade.sol#82-107):
    External call (nextImplementation, data) (contracts/traits/ERC1967Upgrade.sol#42)
        - _functionDelegateCall(nextImplementation, abi.encodeWithSignature("upgradeTo(address)", oldImplementation)) (contracts/traits/ERC1967Upgrade.sol#100)
        - _functionDelegateCall(nextImplementation, abi.encodeWithSignature("oldImplementation()")) (contracts/traits/ERC1967Upgrade.sol#100)
    User call (nextImplementation, data) (contracts/traits/ERC1967Upgrade.sol#105)
        - _upgradeTo(nextImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
        - _upgradeTo(nextImplementation) (contracts/traits/ERC1967Upgrade.sol#105)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/lib/Address.sol#2-36) uses assembly
Address.verifyCallResult(bool,bytes,string) (contracts/lib/Address.sol#165-185) uses assembly
- INLINE ASM (contract/lib/Address.sol#180-183)
EnumerableSet.contains(EnumerableSet.UintSet, uint256) (contracts/lib/EnumerableSet.sol#127-219)
EnumerableSet.values(EnumerableSet.UintSet) (contract/lib/EnumerableSet.sol#194-355) uses assembly
- INLINE ASM (contract/lib/EnumerableSet.sol#193-202)
StorageSlot.getAddressSlot(address) (contract/lib/StorageSlot.sol#52-54)
StorageSlot.getBooleanSlot(bytes32) (contract/lib/StorageSlot.sol#60-64) uses assembly
StorageSlot.getByte32Slot(bytes32) (contract/lib/StorageSlot.sol#68-73) uses assembly
- INLINE ASM (contract/lib/StorageSlot.sol#70-73)
StorageSlot.getU16Slot(bytes32) (contract/lib/StorageSlot.sol#78-82) uses assembly
- INLINE ASM (contract/lib/StorageSlot.sol#80-81)
Context.msgSender() (contracts/traits/Context.sol#16-25) uses assembly
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-use

AccessControl.setRole(bytes32,address) (contracts/traits/AccessControl.sol#122-141) is never used and should be removed
AccessControl.initializeAccessControl() (contracts/traits/AccessControl.sol#145-153) is never used and should be removed
Address.functionCall(address,bytes) (contract/lib/Address.sol#175-181) is never used and should be removed
Address.functionCall(address,bytes,string) (contract/lib/Address.sol#189-195) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contract/lib/Address.sol#122-133) is never used and should be removed
Address.functionStaticCall(address,bytes) (contract/lib/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,uint256) (contract/lib/Address.sol#141-143) is never used and should be removed
Address.sendValue(address,uint256) (contract/lib/Address.sol#154-159) is never used and should be removed
Context.msgData() (contract/traits/Context.sol#127-131) is never used and should be removed
Context.msgValue() (contract/traits/Context.sol#127-131) is never used and should be removed
should be removed
ERC1965Interface.initializeERC1965Interface() (contracts/traits/ERC1965Interface.sol#120-22) is never used and should be removed
ERC1965Interface.registerInterface(bytes) (contracts/traits/ERC1965Interface.sol#142-145) is never used and should be removed
ERC1965Interface.unregisterInterface(bytes) (contracts/traits/ERC1965Interface.sol#146-149) is never used and should be removed
ERC1965Upgrade._changeAdmin(address) (contracts/traits/ERC1967Upgrade.sol#141-148) is never used and should be removed
ERC1965Upgrade._getAdmin(address) (contracts/traits/ERC1967Upgrade.sol#149-152) is never used and should be removed
ERC1965Upgrade._setAdmin(address) (contracts/traits/ERC1967Upgrade.sol#151-154) is never used and should be removed
ERC1965Upgrade._setRole(address,bytes,bytes) (contracts/traits/ERC1967Upgrade.sol#155-157) is never used and should be removed
ERC1965Upgrade._upgradeToAndCall(address,bytes,bool) (contracts/traits/ERC1967Upgrade.sol#166-173) is never used and should be removed
ERC1965Upgrade.initializeERC1967Upgrade() (contracts/traits/ERC1967Upgrade.sol#178) is never used and should be removed
UserDB.addUser(bytes32,address) (contracts/traits/UserDB.sol#1-11) is never used and should be removed
EnumerableSet.at(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#157-159) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#155-157) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (contracts/lib/EnumerableSet.sol#156-158) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#174-176) is never used and should be removed
EnumerableSet.setLength(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#177-179) is never used and should be removed
EnumerableSet.values(EnumerableSet.Bytes32Set) (contracts/lib/EnumerableSet.sol#180-303) is never used and should be removed
EnumerableSet.setValues(EnumerableSet.Bytes32Set,bytes32) (contracts/lib/EnumerableSet.sol#194-202) is never used and should be removed
EnumerableSet.setValues(EnumerableSet.UintSet) (contracts/lib/EnumerableSet.sol#194-355) is never used and should be removed
Pausable.initializePausable() (contracts/traits/Pausable.sol#3-32) is never used and should be removed
StorageSlot.getBooleanSlot(bytes32) (contract/lib/StorageSlot.sol#178-182) is never used and should be removed
StorageSlot.getU16Slot(bytes32) (contract/lib/StorageSlot.sol#178-182) is never used and should be removed
String.equals(string,string) (contracts/lib/Strings.sol#67-71) is never used and should be removed
String.toBytes(string) (contracts/lib/Strings.sol#109-113) is never used and should be removed
String.toBytes32(string) (contracts/lib/Strings.sol#124-128) is never used and should be removed
Upgradable.initializeUpgradable(address) (contracts/traits/Upgradable.sol#16-26) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

```

- The reentrancies flagged in `Marketplace.sol` are false positives although it's recommended to add the `nonReentrant` modifier in all the external functions.
- The unprotected-upgradeable-contract alerts are false positives. Although the contracts should be deployed using a factory pattern to prevent front-running.

THANK YOU FOR CHOOSING
HALBORN