METATRUST

Draft
Security Assessment for

112-2022-04-backd (StaticFail) (1OK-WOC) (1Positive-FLP)

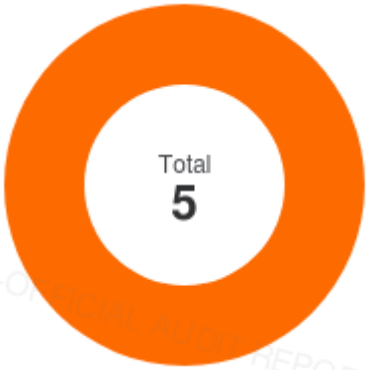
July 14, 2023

Executive Summary

Overview	
Project Name	112-2022-04-backd (StaticFail) (1OK-WOC) (1Positive-FLP)
Codebase URL	https://github.com/code-423n4/2022-04-backd
Scan Engine	AI Analyzer
Scan Time	2023/07/14 22:06:26
Commit Id	c856714

Total	
Critical Issues	0
High risk Issues	5
Medium risk Issues	0
Low risk Issues	0
Informational Issues	0

Critical Issues	The issue can cause large economic losses, large-scale data disorder, loss of control of authority management, failure of key functions, or indirectly affect the correct operation of other smart contracts interacting with it.
High Risk Issues	The issue puts a large number of users' sensitive information at risk or is reasonably likely to lead to catastrophic impacts on clients' reputations or serious financial implications for clients and users.
Medium Risk Issues	The issue puts a subset of users' sensitive information at risk, would be detrimental to the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
Low Risk Issues	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low-impact in view of the client's business circumstances.
Informational Issue	The issue does not pose an immediate risk but is relevant to security best practices or Defence in Depth.



Critical Issues	0%	0
High risk Issues	100%	5
Medium risk Issues	0%	0
Low risk Issues	0%	0
Informational Issues	0%	0

Summary of Findings

MetaScan security assessment was performed on **July 14, 2023 22:06:26** on project **112-2022-04-backd (StaticFail) (1OK-WOC) (1Positive-FLP)** with the repository **https://github.com/code-423n4/2022-04-backd** on branch **default branch**. The assessment was carried out by scanning the project's codebase using the scan engine **AI Analyzer**. There are in total **5** vulnerabilities / security risks discovered during the scanning session, among which **0** critical vulnerabilities, **5** high risk vulnerabilities, **0** medium risk vulnerabilities, **0** low risk vulnerabilities, **0** informational issues.

ID	Description	Severity
MSA-001	Wrong Accounting Order II	High risk
MSA-002	Wrong Accounting Order II	High risk
MSA-003	Wrong Accounting Order II	High risk
MSA-004	Insecure LP Token Value Calculation	High risk
MSA-005	Insecure LP Token Value Calculation	High risk

Findings

Critical (0)

No Critical vulnerabilities found here

High risk (5)

1. Wrong Accounting Order II

High risk Security Analyzer

Invoking user checkpoint should be executed before calculating new balance, share, stake, loan or fee.

File(s) Affected

backd/contracts/StakerVault.sol #105-123

```
105     function transfer(address account, uint256 amount) external override notPaused returns (bool) {
106         require(msg.sender != account, Error.SELF_TRANSFER_NOT_ALLOWED);
107         require(balances[msg.sender] >= amount, Error.INSUFFICIENT_BALANCE);
108 
109         ILiquidityPool pool = controller.addressProvider().getPoolForToken(token);
110         pool.handleLpTokenTransfer(msg.sender, account, amount);
111 
112         balances[msg.sender] -= amount;
113         balances[account] += amount;
114 
115         address lpGauge = currentAddresses[_LP_GAUGE];
116         if (lpGauge != address(0)) {
117             ILpGauge(lpGauge).userCheckpoint(msg.sender);
118             ILpGauge(lpGauge).userCheckpoint(account);
119         }
120 
121         emit Transfer(msg.sender, account, amount);
122         return true;
123     }
```

Recommendation

Check the business logic and move the statements about invoking user checkpoint forward.

2. Wrong Accounting Order II

High risk Security Analyzer

Invoking user checkpoint should be executed before calculating new balance, share, stake, loan or fee.

File(s) Affected

backd/contracts/StakerVault.sol #133-177

```
133     function transferFrom(
134         address src,
135         address dst,
136         uint256 amount
137     ) external override notPaused returns (bool) {
138         /* Do not allow self transfers */
139         require(src != dst, Error.SAME_ADDRESS_NOT_ALLOWED);
140 
141         address spender = msg.sender;
142 
143         /* Get the allowance, infinite for the account owner */
144         uint256 startingAllowance = 0;
145         if (spender == src) {
146             startingAllowance = type(uint256).max;
147         } else {
148             startingAllowance = _allowances[src][spender];
149         }
150         require(startingAllowance >= amount, Error.INSUFFICIENT_BALANCE);
151 
152         uint256 srcTokens = balances[src];
153         require(srcTokens >= amount, Error.INSUFFICIENT_BALANCE);
154 
155         address lpGauge = currentAddresses[_LP_GAUGE];
156         if (lpGauge != address(0)) {
157             ILpGauge(lpGauge).userCheckpoint(src);
158             ILpGauge(lpGauge).userCheckpoint(dst);
159         }
160         ILiquidityPool pool = controller.addressProvider().getPoolForToken(token);
161         pool.handleLpTokenTransfer(src, dst, amount);
162 
163         uint256 allowanceNew = startingAllowance - amount;
164         uint256 srcTokensNew = srcTokens - amount;
165         uint256 dstTokensNew = balances[dst] + amount;
166 
167         /* Update token balances */
168         balances[src] = srcTokensNew;
169         balances[dst] = dstTokensNew;
170 
171         /* Update allowance if necessary */
172         if (startingAllowance != type(uint256).max) {
173             _allowances[src][spender] = allowanceNew;
174         }
175         emit Transfer(src, dst, amount);
176         return true;
177     }
```

Recommendation

Check the business logic and move the statements about invoking user checkpoint forward.

3. Wrong Accounting Order II

High risk Security Analyzer

Invoking user checkpoint should be executed before calculating new balance, share, stake, loan or fee.

File(s) Affected

backd/contracts/StakerVault.sol #360-399

```
360     function unstakeFor(
361         address src,
362         address dst,
363         uint256 amount
364     ) public override returns (bool) {
365         ILiquidityPool pool = controller.addressProvider().getPoolForToken(token);
366         uint256 allowance_ = _allowances[src][msg.sender];
367         require(
368             src == msg.sender || allowance_ >= amount || address(pool) == msg.sender,
369             Error.UNAUTHORIZED_ACCESS
370         );
371         require(balances[src] >= amount, Error.INSUFFICIENT_BALANCE);
372         address lpGauge = currentAddresses[_LP_GAUGE];
373         if (lpGauge != address(0)) {
374             ILpGauge(lpGauge).userCheckpoint(src);
375         }
376         uint256 oldBal = IERC20(token).balanceOf(address(this));
377 
378         if (src != dst) {
379             pool.handleLpTokenTransfer(src, dst, amount);
380         }
381 
382         IERC20(token).safeTransfer(dst, amount);
383 
384         uint256 unstaked = oldBal - IERC20(token).balanceOf(address(this));
385 
386         if (src != msg.sender && allowance_ != type(uint256).max && address(pool) != msg.sender) {
387             // update allowance
388             _allowances[src][msg.sender] -= unstaked;
389         }
390         balances[src] -= unstaked;
391 
392         if (strategies[src]) {
393             strategiesTotalStaked -= unstaked;
394         } else {
395             _poolTotalStaked -= unstaked;
396         }
397         emit Unstaked(src, amount);
398         return true;
399     }
```

Recommendation

Check the business logic and move the statements about invoking user checkpoint forward.

4. Insecure LP Token Value Calculation

High risk Security Analyzer

Liquidity token value/price can be manipulated to cause flashloan attacks.

File(s) Affected

backd/contracts/pool/LiquidityPool.sol #506-536

```
506     function depositFor(
507         address account,
508         uint256 depositAmount,
509         uint256 minTokenAmount
510     ) public payable override notPaused returns (uint256) {
511         uint256 rate = exchangeRate();
512 
513         if (isCapped()) {
514             uint256 lpBalance = lpToken.balanceOf(account);
515             uint256 stakedAndLockedBalance = staker.stakedAndActionLockedBalanceOf(account);
516             uint256 currentUnderlyingBalance = (lpBalance + stakedAndLockedBalance).scaledMul(rate);
517             require(
518                 currentUnderlyingBalance + depositAmount <= depositCap,
519                 Error.EXCEEDS_DEPOSIT_CAP
520             );
521         }
522 
523         _doTransferIn(msg.sender, depositAmount);
524         uint256 mintedLp = depositAmount.scaledDiv(rate);
525         require(mintedLp >= minTokenAmount, Error.INVALID_AMOUNT);
526 
527         lpToken.mint(account, mintedLp);
528         _rebalanceVault();
529 
530         if (msg.sender == account || address(this) == account) {
531             emit Deposit(msg.sender, depositAmount, mintedLp);
532         } else {
533             emit DepositFor(msg.sender, account, depositAmount, mintedLp);
534         }
535         return mintedLp;
536     }
```

backd/contracts/pool/LiquidityPool.sol #111-120

```
111     function depositAndStake(uint256 depositAmount, uint256 minTokenAmount)
112         external
113         payable
114         override
115         returns (uint256)
116     {
117         uint256 amountMinted_ = depositFor(address(this), depositAmount, minTokenAmount);
118         staker.stakeFor(msg.sender, amountMinted_);
119         return amountMinted_;
120     }
```

Recommendation

Do not use AMM pool or custom liquidity calculation to caculate LP token value/price.

5. Insecure LP Token Value Calculation

High risk Security Analyzer

Liquidity token value/price can be manipulated to cause flashloan attacks.

File(s) Affected

backd/contracts/pool/LiquidityPool.sol #544-570

```
544     function redeem(uint256 redeemLpTokens, uint256 minRedeemAmount)
545         public
546         override
547         returns (uint256)
548     {
549         require(redeemLpTokens > 0, Error.INVALID_AMOUNT);
550         ILpToken lpToken_ = lpToken;
551         require(lpToken_.balanceOf(msg.sender) >= redeemLpTokens, Error.INSUFFICIENT_BALANCE);
552 
553         uint256 withdrawalFee = addressProvider.isAction(msg.sender)
554             ? 0
555             : getWithdrawalFee(msg.sender, redeemLpTokens);
556         uint256 redeemMinusFees = redeemLpTokens - withdrawalFee;
557         // Pay no fees on the last withdrawal (avoid locking funds in the pool)
558         if (redeemLpTokens == lpToken_.totalSupply()) {
559             redeemMinusFees = redeemLpTokens;
560         }
561         uint256 redeemUnderlying = redeemMinusFees.scaledMul(exchangeRate());
562         require(redeemUnderlying >= minRedeemAmount, Error.NOT_ENOUGH_FUNDS_WITHDRAWN);
563 
564         _rebalanceVault(redeemUnderlying);
565 
566         lpToken_.burn(msg.sender, redeemLpTokens);
567         _doTransferOut(payable(msg.sender), redeemUnderlying);
568         emit Redeem(msg.sender, redeemUnderlying, redeemLpTokens);
569         return redeemUnderlying;
570     }
```

backd/contracts/pool/LiquidityPool.sol #435-449

```
435     function unstakeAndRedeem(uint256 redeemLpTokens, uint256 minRedeemAmount)
436         external
437         override
438         returns (uint256)
439     {
440         uint256 lpBalance_ = lpToken.balanceOf(msg.sender);
441         require(
442             lpBalance_ + staker.balanceOf(msg.sender) >= redeemLpTokens,
443             Error.INSUFFICIENT_BALANCE
444         );
445         if (lpBalance_ < redeemLpTokens) {
446             staker.unstakeFor(msg.sender, msg.sender, redeemLpTokens - lpBalance_);
447         }
448         return redeem(redeemLpTokens, minRedeemAmount);
449     }
```

Recommendation

Do not use AMM pool or custom liquidity calculation to caculate LP token value/price.

Medium risk (0)

No Medium risk vulnerabilities found here

Low risk (0)

No Low risk vulnerabilities found here

Informational (0)

No Informational vulnerabilities found here

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without MetaTrust's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts MetaTrust to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. MetaTrust's position is that each company and individual are responsible for their own due diligence and continuous security. MetaTrust's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by MetaTrust is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS 112-2022-04-backd (StaticFail) (1OK-WOC) (1Positive-FLP) Security Assessment AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER

APPLICABLE LAW, MetaTrust HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, MetaTrust SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, MetaTrust MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, MetaTrust PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER MetaTrust NOR ANY OF MetaTrust'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. MetaTrust WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT MetaTrust'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING 112-2022-04-backd (StaticFail) (1OK-WOC) (1Positive-FLP) Security Assessment MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST MetaTrust WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF MetaTrust CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST MetaTrust WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.