STAX EXPLOIT INCIDENT REPORT

1.1 Overview

What happened
At 9:11am EST, A total of 321,154 xLP tokens were taken from the xLP Staking contract. These tokens were swapped for precisely 1,418,303 $TEMPLE and 1,262,438 $FRAX. 1,418,303 $TEMPLE was then sold for 1,116,243 FRAX.
https://etherscan.io/tx/0x8c3f442fc6d640a6ff3ea0b12be64f1d4609ea94edd2966f42c01cd9bdcf04b5

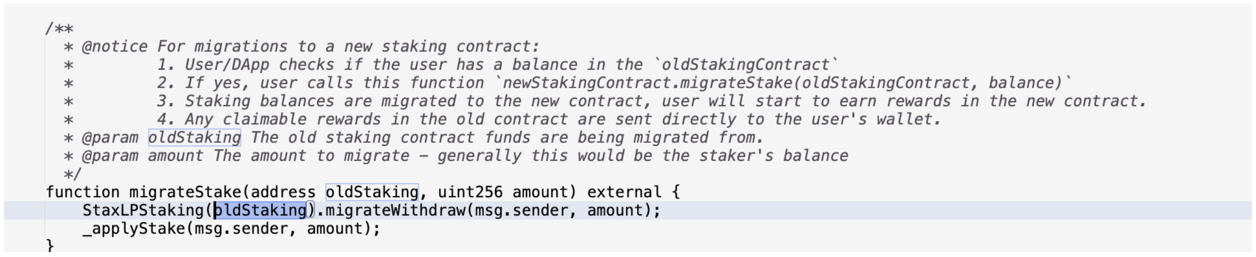
Only one malicious agent was involved. This is the address driving the exploit:
https://etherscan.io/address/0x2b63d4a3b2db8acbb2671ea7b16993077f1db5a0

Then, around 2:28 pm EST, the team identified there might be a second facet to the exploit that allows someone to claim additional FXS rewards.

This issue was quickly identified and mitigated limiting any potential loss to 285 FXS at risk (approx. $1.5k) of FXS at the time it was identified.

https://etherscan.io/address/0xbe78d20adc3e1fc641c2f8d6e9fcdf50bd63901d
https://etherscan.io/address/0x000000000000df8c944e775bde7af50300999283

How did it happen
The STAX staking contract had a method to allow users to migrate LP from the older staking contract to the new one.

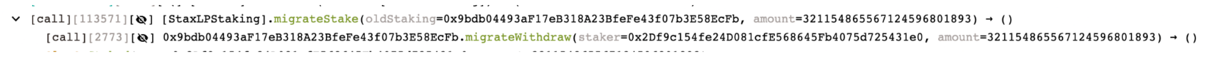


```
/**
 * @notice For migrations to a new staking contract:
 *         1. User/DApp checks if the user has a balance in the `oldStakingContract`
 *         2. If yes, user calls this function `newStakingContract.migrateStake(oldStakingContract, balance)`
 *         3. Staking balances are migrated to the new contract, user will start to earn rewards in the new contract.
 *         4. Any claimable rewards in the old contract are sent directly to the user's wallet.
 * @param oldStaking The old staking contract funds are being migrated from.
 * @param amount The amount to migrate - generally this would be the staker's balance
 */
function migrateStake(address oldStaking, uint256 amount) external {
    StaxLPStaking(oldStaking).migrateWithdraw(msg.sender, amount);
    _applyStake(msg.sender, amount);
}
```

Any external contract call on an uncontrolled address is a potential attack vector, and this should have had some validation/access control. Despite correctly managing this in other instances across contracts, unfortunately, this method on this contract did not.

Therefore, the exploiter created their own staking contract and passed it to the method. Their contract effectively

1. Had an empty Implemented 'migrateWithdraw' (that is, made it a no-op)
2. Called migrateStake with the max possible stake, thereby swamping the staking pool



```
⌄ [call][113571][] [StaxLPStaking].migrateStake(oldStaking=0x9bdb04493aF17eB318A23BfeFe43f07b3E58EcFb, amount=321154865567124596801893) → ()
    [call][2773][] 0x9bdb04493aF17eB318A23BfeFe43f07b3E58EcFb.migrateWithdraw(staker=0x2Df9c154fe24D081cfE568645Fb4075d725431e0, amount=321154865567124596801893) → ()
```

This allowed the attacker to increase their staking balance in the staking contract. The expoiter proceeded to drain the LP funds, and liquidate to ETH.
https://etherscan.io/tx/0x4b119a4f4ba1ad483e9851973719f310527b43f3fcc827b6d52db9f4c1ddb6a2

The attacker repeated the hack to try and drain the FXS and TEMPLE rewards from the contract so they could own as much as the staking pool as possible. By staking an infinite amount, they now own 99.99% of the stake, they can now claim any rewards for the coming 3 days (~ 1.5k USD Notional of FXS).

1.2 Impact

<u>What was impacted</u>

STAX was drained of xLP equalling $2.4M (321,154 xLP tokens) in funds being stolen. These tokens were swapped for precisely 1,418,303 $TEMPLE and 1,262,438 $FRAX. 1,418,303 $TEMPLE was then sold for 1,116,243 FRAX. STAX v0.1 had a total of 16 active users, all of which were affected by this exploit.

Attacker now controls 99.9999% of the staking pool, so they have a near 100% claim on the 285 FXS of emissions this week.

For any new xLP deposited, it would have been an arms race between the team, existing stakers and the attacker as to who withdrew it first. To mitigate this, we took down the dApp.

Temple DAO will make affected users whole via DAO Funds.

What wasn't impacted
Temple CORE Vault contracts share no common code with STAX, have been audited by PeckShield, and remain secure. Audit results can be reviewed here:
https://docs.templedao.link/technical-reference/audits

From a STAX perspective, user-earned but unclaimed rewards (up to the exploit) are still available and can be claimed. Since the dApp is down, these rewards cannot be claimed through the front-end. They can still be claimed directly through the contract and the team would be happy to assist any affected users with the steps should they prefer to claim prior to us re-launching the dApp. The dApp will be re-launched once an audit has been completed.

The STAX liquidityOps contracts, which are locked into the TEMPLE/FRAX gauge, are also unaffected. Lock details

- via Convex:
    - LP: 16322.2493815572
    - unlock ts: 1708091226
    - Fri Feb 16 2024 13:47:06 GMT+0000
- via direct lock:
    - LP: 436318.410527874
    - unlock ts: 1700912220
    - Sat Nov 25 2023 11:37:00 GMT+0000

1.3 Root Cause

Regarding contract development, we have an ethos of avoiding upgradeable contracts where possible. Once the game is set, people are in. But at times and particularly in early stages of the protocol we allow some flexibility for evolution or to be able to fix problems.

How do we weigh that balance? If there is a new version we want the people to be able to migrate over without huge pain and cost to the user. That is what the method that was exploited (Migrate Stake) allows - to go from one staking contract to another.

The critical mistake is one of access control. Because that method was public, we need control where the user migrates to. Without access control the exploiter set their balance of xLP falsely high, and used this to spoof the migrator into giving them all of the xLP.

While we have properly managed this in other contracts and methods before, in this case, despite having two peer reviews, testing, and significant time on test net, it fell through the cracks. It was a simple error that should never have occurred.

1.4 Process used to develop the contract that was exploited

Original author wrote the contract. Then:

1. We had code review from 2 engineers
2. We had extensive tests, which focused on the user facing public methods
3. We had a long testing period in testnet

This is considered sufficient practice and no obvious holes in our process were identified at a high level.

This bug is primarily introduced through simple human error. In over 10,000 lines of contract code developed through Temple and STAX, this is expected. But having the error make it all the way into production is not expected.

We seek to increase the quality and intensity of review on all contracts. Aside from simple human error, this contributed to the exploit being missed as it was an alpha release, a non-core migration contract, and audits were not in place pre- full launch due to expectation of frequent iteration.

1.5 Process improvements to reduce likelihood of future exploit

Many changes and improvements are underway.

1. Complete code review of stax and temple for this exploit in other places (done)
2. Complete code review of stax and temple from all other attack vectors (in process)
3. Implement formal review checklist for all future reviews
4. Make one reviewer with clear responsibility for review
5. Increase full scope of testing on a broader scope of contracts (incl anything that calls outside of system code)
6. Get more frequent audits even with lower tier (faster/easier to access) firms for any contract in the broader scope of potentially at risk contracts
7. Add a static checker to automated CI

1.6 Timeline

Time	Person	Action
10/11/22 9:11 am EST	exploiter	Exploit occured https://etherscan.io/tx/0x4b119a4f4ba1ad483e9851973719f310527b43f3fcc827b6d52db9f4c1ddb6a2
10/11/22 9:12 am EST	mirionic	Links provided that STAX may be experiencing an exploit. Investigation as to what happened on chain and the contracts affected with assistance from princetonbishop, dennet, and smokey
10/11/22 9:46 am EST	dariox	Confirmation that core is not at risk
10:04 am EST	dennet	Investigation on if this is an issue with oldStaking.
10:10 am EST	dennet	Partial root cause: The method inputs should have been validated https://etherscan.io/address/0xd2869042E12a3506100af1D192b5b04D65137941
10/11/22 10:25 am EST	zelki	Front end of stax.fi was taken down to prevent any further deposits
10/11/22 10:29 am EST	doc	Account reported to Binance with assistance from Palladin
10/11/22 10:50 am EST	doc	Binance replies that the account was used by SimpleSwap exchange where user converted from XMR -> ETH. Swap used TOR obfuscating IP links.
10/11/22 10:59 am EST	decen	Initiated discussions with HatsFinance
10/11/22 2:32 pm EST	dariox	Suspicious tx noted: https://etherscan.io/address/0xbe78d20adc3e1fc641c2f8d6e9fcdf50bd63901d And bot sucking up rewards: https://etherscan.io/address/0x000000000000df8c944e775bde7af50300999283 Investigation started
10/11/22	smokey	Investigation uncovered that they are trying to drain the

2:41 pm EST		fxs and temple rewards from the contract by creating a lot of new LP tokens to get a big proportional reward of the tokens
10/11/22 2:51 pm EST	smokey	https://etherscan.io/address/0xd2869042e12a3506100af1d192b5b04d65137941/advanced#tokentxns Can track drainage there
10/11/22 3:04 pm EST	princetonbishop	The exploiter was trying to increase their stakes in the staking contract by repeating the hack, so that they can drain the contract when xLP tokens are sent there. As long as we don't send tokens to the staking contract, they can't drain them. Locker Proxy contract directly sends xLP to the staking contract. So long as no new users are staking, there won't be xLP tokens to drain. For the fxs and temple tokens, they can't use this method to collect those tokens.
10/11/22 3:15 pm est	dennet	hackers max staking in order to drain rewards tokens (fxs,temple) gradually
10/11/22 3:37 pm EST	Frontier and lux	removing minting rights
10/11/22 3:46 pm EST	butlerji	Suggestion to stop liquidityOps to be on the safe side (msig we 100% control) Err on the side of overstepping here. Pause all minting, then we can carefully re-enable after a triple check review
10/11/22 3:49 pm EST	frontier	Remove liquidityOps from msig
10/11/22 4:12 pm EST	smokey	Smokey replicated the hack with their own contract locally. We weren’t sure if we needed this in our own defense. Thankfully, in the end, we didn’t
10/11/22	butlerji	Assess the way we drain FXS Rewards safely

4:19 pm EST		
10/11/22 4:48 pm EST	butlerji	Confirmed that no more xLP minting is possible
10/11/22 5:02 pm EST	frontier	Update rewards manager to multisig so future harvested rewards are sent directly to multisig https://etherscan.io/tx/0xc61ea0907cde628a10b47f51bd696d9f1a5445be8bb68a06314d1a4ecb5f857a
10/11/22 5:26 pm EST	dennet	Confirmed that the exploiter could get his hands on 285 FXS = $1430. There were 667 FXS deposited 4 days ago. He can claim 3/7 of that, because 3 days left.
10/11/22 5:54 pm EST	frontier	Re-directed frax gauge reward harvesting to the stax multisig, instead of the staking contract.
10/11/22 9:48 pm EST	alfa	Reported exploiters address (holding funds) to etherscan to be marked as exploiter wallet
10/12/22 4:32 am EST	Butlerji and frontier	Confirmation that the exploit was not present anywhere else in the code for both STAX and TEMPLE

1.7 Next Actions

Immediate Action

Action	Owner	Progress	Comments
Bring down STAX site	Zelki	done	
Announcement(s) on Discord	Doc	done	
Announcement(s) on Twitter	Saint Alfa	done	
Gather wallets of STAX exploit victims	Dariox, Bobruisk, Makaveli	done	
Root Cause Analysis	frontier	done	

Audit and analyze existing stax and temple code/contracts	Frontier and butler	done	
Publish incident report	doc	done	
HatsFinance Bounty	decen	Done	Bounty deployed
Implement process improvements in section 2.4	Butlerji	done	
Repay affected users		done	