SOLID GROUP

X

Security Assessment

June 30rd, 2021

For:
Secure Pad

Disclaimer

Solid Group reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. Solid group do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc’…)

Solid Group Audits do not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analysed, nor do they provide any indication of the technology proprietors. Solid Group Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

Soli Group Reports represent an extensive auditing 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.

Solid Group hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Solid Group, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption)

Solid Group’s position is that each company and individual are responsible for their own due diligence and continuous security.

Solid Group in no way claims any guarantee of security or functionality of the technology we agree to analyze.

SOLID GROUP

About Secure Pad

Secure Pad is a launchpad, with a central focus on safety for crypto investors, while ensuring a successful fundraise & launch for new projects. $SEPA token holders participating in IDO's launched through our platform, gain added protection through our Insurance Fund. The future for Secure Pad will also include a Decentralized Incubator, which will aim to pair project Founders with vetted, skilled, & competent industry professionals in the network. Secure Pad's goal is to help bring Founder's vision from idea to working product, while giving $SEPA token holders exclusive early access to their IDO's.

🌍 Website: http://securepad.io/
💬 Telegram: https://t.me/SecurePad
📣 Telegram ANN: https://t.me/SecurePadAnnouncements
📟 Twitter: https://twitter.com/secure_pad
📘 Whitepaper: https://securepad.io/Sepa-Whitepaper.pdf

About Solid Group

Solid Group is a blockchain consulting and auditing service provider, founded by 3 cybersecurity experts with a passion for thinking out of the box, learning, and sharing knowledge. Every project goes through a meticulous process and is viewed by at least two partners, thereby achieving a high level of credibility and professionalism. Our group is partnered with multiple organizations and launchpads that have a combined market cap of over 400 million USD.

Telegram | Telegram discussion group | Twitter | Contact for audit | Audit Checker bot | Medium

Files in Scope

Repositories	Commit Final Revision
https://github.com/extrawatts/sepa-securepad	3ac5935461c0e906dd8f0bb66ee6a634d59a1221
https://github.com/extrawatts/sepa-liquidity	be6576d1deebc7abe582585217763271b932ed07
https://github.com/extrawatts/ securepad-crowdsale	134533391d0c4849d67c9ca2ddc32f43492ba5cb

Findings

● Informational severity Issues	4
● Low severity issues	1
● Medium severity issues	1
● High severity issues	0

Sepa-SecurePad Findings

Repository	Commit	Files in Scope
https://github.com/extrawatts/sepa-securepad	920db2fe34ecd458e4bb96ef4cd6b325278078ca	Crowdsale.sol CrowdsaleFactory.sol Mock/Farm.sol Pool.sol
https://github.com/extrawatts/sepa-securepad	3ac5935461c0e906dd8f0bb66ee6a634d59a1221	Crowdsale.sol CrowdsaleFactory.sol Mock/Farm.sol Whitelist.sol

Crowdsale.sol

Issue #1

Type	Severity	Location	Status
Best Practice	● Medium	buyTheRest()	✅ Fixed

Description

The number of tokens that were left is not being updated correctly.

Recommendation

Our recommendation is to update the following values as follows:

```
tokenLeft -= rewardedAmount;

soldAmount += rewardedAmount;

totalRaise += value;

participants += 1
```

Issue #2

Type	Severity	Location	Status
Logical Issue	Informational	buy()	Not Fixed

Description

The number of participants is calculated incorrectly, the team has ignored a scenario when a holder participate more than once.

Sepa-liquidity Findings

Repository	Commit	Files in Scope
https://github.com/extrawatts/sepa-liquidity	422fe67f259076b9253ea85bc01532c1477a1567	LiquidityMining.sol LiquidityMiningFactory.sol
https://github.com/extrawatts/sepa-liquidity	be6576d1deebc7abe582585217763271b932ed07	LiquidityMining.sol LiquidityMiningFactory.sol

LiquidityMining.sol

Issue #1:

Type	Severity	Location	Status
Gas Optimization	Informational	Initialize function	Not Fixed

Description
rewardToken, stakingToken, initreward, startime, DURATION variables are only set once on initialization. They can be declared as immutable to save gas.

Recommendation
Values that are only set once on initialization can be declared as immutable to save gas on every future access to them.
This does require splitting the current initialize() function into a constructor that takes care of the variable initializations, but since it will save gas on all future calls, it may be worth it.

Issue #2:

Type	Severity	Location	Status
Best Practice	Informational	emergencyWithdraw	❌ Not Fixed

Description
The emergencyWithdraw function and _withdraw function are similar except for a single require line

Recommendation
Consider refactoring emergencyWithdraw so that it will reuse the code in _withdraw.

Issue #3:

Type	Severity	Location	Status
Gas Optimization	Informational	stake	✅ Fixed

Description
The stake function can be declared as external as it isn't used from inside the contract

Recommendation
Consider declaring it as external to save gas

Issue #4:

Type	Severity	Location	Status
Volatile Code	Potentially High	stake	✓ acknowledged

Description
We didn't audit the SEPA token or any other token secure-pad team plans to deposit to their staking platform. However, there is a potentially high severity issue that is worth mentioning. The current code may suffer from a high severity issue if the staking token will be token with fees on transfer.

Every time a user deposits tokens to the contract, the contract receives a little bit less due to the fee. The problem is that the user can still withdraw the original amount (including the fees).

Recommendation
Our recommendation is to add support for tokens with transfer fees.

The team acknowledged the issue and commented that there are no fees in their token code when interacting with the staking contract.

Sepa-Crowdsale Findings

Repository	Commit	Files in Scope
https://github.com/extrawatts/securepad-crowdsale	7c698b32b731dfa11ba4a4e97b34b9e38f4c9f77	Farm.sol Authorizable.sol
https://github.com/extrawatts/securepad-crowdsale	134533391d0c4849d67c9ca2ddc32f43492ba5cb	Farm.sol Authorizable.sol

Farm.sol

Issue #1:

Type	Severity	Location	Status
Logical Issue	🔴 High	initialize	✅ Fixed

Description

Calling initialize() with the same reward token multiple times overrides the previous parameters of the token.

Recommendation

require that the token isn't in the list.

Issue #2:

Type	Severity	Location	Status
Volatile Code	🟡 Medium	Farm.sol	❌ Not Fixed

Description

There are multiple places in the contract that iterates over an unbounded array (tokenAddresses). This can potentially exceed the block's gas limit and make transactions fail.
Also note that the tokenAddresses array is always getting bigger - there's no logic that removes elements from the array.

Recommendation

Consider limiting the size of the array or the size of the iteration.

Issue #3:

Type	Severity	Location	Status
Gas Optimization	Informational	Farm.sol	✅ Fixed

Description

variables that are only set once in the constructor (such as total and fixedConstantPerToken) can be declared immutable to save gas.

Issue #4:

Type	Severity	Location	Status
Owner Capabilities	Low	giveaway()	❌ Not Fixed

Description

The giveAway function gives the admins complete control over the points balance of any address.

Recommendation

Consider transferring this capability to a contract or use a more specific role.

Issue #5:

Type	Severity	Location	Status
Owner Capabilities	Low	farmed, farmedStart, getConsolidatedRewards	✅ Fixed

Description

These functions can be declared as external to save gas Recommendation

Issue #6:

Type	Severity	Location	Status
Owner Capabilities	Low	payment	✅ Fixed

Description

This error message is inaccurate.

```
require(st.points > 0, "amount equal to 0");
```

Recommendation

Rephrase the error message

```
require(st.points > 0, "accrued points equal to 0")
```

Issue #7:

Type	Severity	Location	Status
Logical Issue	High	withdraw	✅ Fixed

Description

If the user isn't eligible for a reward on a single reward token (e.g because he called getReward manually), the withdraw function will always fail.

Recommendation

use try-except to gracefully handle errors that shouldn't cause the main flow to error out

Issue #8:

Type	Severity	Location	Status
Best Practice	Medium	consolidate	✅ Fixed

Description

Part of the side effects of consolidate function is to update the staker's timestamp.

Recommendation

Update the staker's timestamp in the consolidate function, instead of manually after every call to consolidate.

Issue #9:

Type	Severity	Location	Status
Logical Issue	Medium	getReward	✅ Fixed

Description

If the user isn't eligible for a reward on a single reward token (e.g because he called getReward manually), the withdraw function will always fail.

Recommendation

use try-except to gracefully handle errors that shouldn't cause the main flow to error out

Issue #10:

Type	Severity	Location	Status
Logical Issue / Gas Optimization	Informational	remove	❌ Not Fixed

Description

The remove function removes an element from an array by iterating over all elements (O(n) work), while it can be done in O(1) - simply swap the element to be removed with the last element and call pop.