Swiss Stake

Security Assessment (Summary)

March 6, 2020

Prepared For:
Michael Egorov | *Swiss Stake*
michael@swiss-stake.com

Prepared By:
Josselin Feist | *Trail of Bits*
josselin@trailofbits.com

Assessment Summary

During the week of January 13 to January 17, 2020, Trail of Bits performed an assessment of the Curve.fi smart contract (2c7494a1) with one engineer. Trail of Bits reported seven issues and one code quality recommendation. On February 18, Trail of Bits reviewed the fixes made to reported vulnerabilities from commit hash 65365742.

Throughout this assessment, Trail of Bits sought to answer various questions about the security of the contract. We focused on flaws that would allow an attacker to:

- Drain the contract's funds
- Prevent liquidity providers from accessing their funds
- Break the pool's invariant

We performed manual code analysis and property-based fuzzing with Echidna. Several issues found were the results of fuzzing. The properties manually or automatically checked include:

- Can the arithmetic rounding be abused to drain the contract?
- Are the front-running risks well understood?
- Are Vyper language-specific issues present?
- Can the pool's creator access unexpected privileges?
- Can the pool be trapped due to gas limitations?
- Are the interactions with the external tokens properly done?
- Is the contract free of re-entrancies?
- Are the functions working with the underlying tokens in the same way as the original functions?
- Can a user pay a greater price than expected?

Due to time constraints, some areas were only partly covered, including the impact of the arithmetic rounding over multiple transactions; our preliminary results showed a negligible impact.

Swiss Stake correctly fixed the reported issues. Trail of Bits made the following additional recommendations:

- Integrate fuzzing as part of the development process and constantly fuzz the pool with parameters close to the deployed values.
- Consider verifying the contract's invariants with symbolic execution.
- Add asserts for the maximum value of admin-controlled parameters.
- Be aware of the gas limitation for future code updates using `get_D`.

- Consider renaming some variables (`PRECISION` -> `LENDING_PRECISION`, `tethered` -> `ERC20NoReturn`).
- Document the case of token migration.
- Consider using parentheses for all arithmetic operations.