Dedaub

Security Technology for Smart Contracts

Maple Finance - Maple Token

Smart Contract Security Assessment

Date: Mar. 12, 2021

Abstract

Dedaub was commissioned to perform a security audit on Maple Finance's MPL token smart contracts. The audit was performed on commit `2a7ba59c355d6b31cc0d251a5f967b21753f566`.

Two auditors worked on this task. We reviewed the code in significant depth, assessed the economics of the protocol and processed it through automated tools. We also decompiled the code and analyzed it, using our static analysis (incl. symbolic execution) tools, to detect possible issues.

Setting and Caveats

The audit was conducted in parallel, and should be considered together, with our audit of the maple-core module. The two codebases share code (almost the entire MPL token functionality is used in the FDT token variants of maple-core).

The token has finite minting, built into its constructor. 10M MPL are ever minted. Burn functionality is included but is unused. The documentation clearly states how the tokens are to be used (in a Balancer pool). This constitutes an element of trust outside the smart contracts themselves.

Vulnerabilities and Functional Issues

This section details issues that affect the functionality of the contract. Dedaub generally categorizes issues according to the following severities, but may also take other considerations into account such as impact or difficulty in exploitation:

Category	Description
Critical	Can be profitably exploited by any knowledgeable third party attacker to drain a portion of the system's or user's funds.
High	Third party attackers may block the system or cause the system or users to lose funds.

Medium	Examples: 1) User or system funds can be lost when third party systems misbehave. 2) DoS, under specific conditions.
Low	Examples: 1) Breaking important system invariants, but without apparent consequences. 2) Buggy functionality for trusted users where a workaround exists. 3) Security issues which may manifest when the system evolves.

Issue resolution includes “dismissed”, by the client, or “resolved”, per the auditors.

Critical Severity

[No critical severity issues]

High Severity

[No high severity issues]

Medium Severity

[No medium severity issues]

Low Severity

Description	Status
The ERC20 `transfer` operations in `ERC2222` will fail, if the underlying `fundToken` is not fully compliant with the ERC20 standard, as it is possible that they do not return a boolean value to indicate the success of the call (most notable exception is USDT).	**Dismissed: MPL token `fundsToken` will always be USDC which is fully compliant.**

It is recommended that the OpenZeppelin SafeERC20 wrappers be used, to ensure compatibility with such tokens.	

Other/Advisory Issues

This section details issues that are not thought to directly affect the functionality of the project, but we recommend addressing.

Description	Status
The `onlyFundsToken` modifier in `MapleToken` is dead code. It is suggested that the unused modifier be removed.	**Resolved**
The contracts were compiled with the Solidity compiler `v0.6.11` which has some known minor issues (but relatively few, compared to earlier versions). We have reviewed the issues and do not believe them to affect the contract. More specifically, at the time of writing, there are 2 known compiler bugs associated with the Solidity compiler `v0.6.11`: • Copying an empty `bytes` or `string` array from memory to storage can cause data corruption: ○ This could affect the `name` and `symbol` fields (without leading to any exploits). Assuming that the Maple Token contract is deployed correctly, with a non-empty name and symbol, this should not be an issue. • Direct assignments of storage arrays with an element size <= 16 bytes (more than one values fit in one 32 byte word) are not correctly cleared if the length of the newly assigned value is smaller than the length of the previous one. (No such array is ever stored.)	**Resolved**

Disclaimer

The audited contracts have been analyzed using automated techniques and extensive human inspection in accordance with state-of-the-art practices as of the date of this report. The audit makes no statements or warranties on the security of the code. On its own, it cannot be considered a sufficient assessment of the correctness status of the contract. While we have conducted an analysis to the best of our ability, it is our recommendation for high-value contracts to commission several independent audits, as well as a public bug bounty program.

About Dedaub

Dedaub offers technology and auditing services for smart contract security. The founders, Neville Grech and Yannis Smaragdakis, are top researchers in program analysis. Dedaub's smart contract technology is demonstrated in the contract-library.com service, which decompiles and performs security analyses on the full Ethereum blockchain.