SHERLOCK

Security Review For Maple

Collaborative Audit Prepared For:	Maple
Lead Security Expert(s):	0x73696d616f
	vinica_boy
Date Audited:	September 8 - September 10, 2025

Introduction

Maple is redefining asset management for the digital age – onchain, transparent, and built to scale. Through institutional-grade lending and yield strategies, Maple brings the rigor of traditional finance to the transparency and innovation of crypto. This is asset management, evolved for the future, global, onchain financial system.

Scope

Repository: maple-labs/globals-v2

Audited Commit: a902c0bec8254e00140aabc1cf369558d4c971ff

Final Commit: c19ed92f23af4903a0d2781f523394ea57936c13

Files:

- contracts/GovernorTimelock.sol
- contracts/interfaces/IGovernorTimelock.sol

Repository: maple-labs/maple-core-v2

Audited Commit: 9b9acdadfe251af87f403c6e47183edc8464c63c

Final Commit: 714f192841da979c8415d52149a1b9a24da0424f

Files:

- scripts/GovernorTimelockDeployment.s.sol

Findings

Each issue has an assigned severity:

- Medium issues are security vulnerabilities that may not be directly exploitable or may require certain conditions in order to be exploited. All major issues should be addressed.
- High issues are directly exploitable security vulnerabilities that need to be fixed.
- Low/Info issues are non-exploitable, informational findings that do not pose a security risk or impact the system's integrity. These issues are typically cosmetic or related to compliance requirements, and are not considered a priority for remediation.

Issues Found

High	Medium	Low/Info
0	0	3

Issues Not Fixed and Not Acknowledged

High	Medium	Low/Info
0	0	0

Issue L-1: Delay can not be set to default value once set to a specific per function value

Source: https://github.com/sherlock-audit/2025-09-maple-sept-8th/issues/4

Summary

The desired feature as per the docs of setting the timelock to 0 to signal the use of the default timelock is not possible once set to per function:

> Contract will allow setting timelock to 0 for specific function selector which means that contract will fallback to using default timelock config in future for that function

Vulnerability Detail

The setter for the function specific timelock is:

```
function setFunctionTimelockParameters(
    address target_,
    bytes4  functionSelector_,
    uint32  delay_,
    uint32  executionWindow_
)
    external override onlySelf
{
    require(delay_           >= MIN_DELAY,            "GT:SFTP:INVALID_DELAY");
    require(executionWindow_ >= MIN_EXECUTION_WINDOW,
    ↪  "GT:SFTP:INVALID_EXEC_WINDOW");

    functionTimelockParameters[target_][functionSelector_] = TimelockParameters({
    ↪  delay: delay_, executionWindow: executionWindow_ });

    emit FunctionTimelockSet(target_, functionSelector_, delay_, executionWindow_);
}
```

As can be seen, delay has always to be >= `MIN_DELAY`, making it impossible to set it to 0 again and allow the usage of the default timelock.

Impact

Timelock of a function can't be set back to the default value.

Code Snippet

https://github.com/sherlock-audit/2025-09-maple-sept-8th/pull/1/files#diff-8976e5067f2102ea34580e0de5ec6762a73d214088306194672c326512646a77R118

Tool Used

Manual Review

Recommendation

Allow setting the delay to 0.

Discussion

lpetroulakis

The team has acknowledged and fixed the issue in https://github.com/maple-labs/globals-v2/commit/c19ed92f23af4903a0d2781f523394ea57936c13

Issue L-2: CEI pattern is not followed in `GovernorTimelock::executeProposals()`

Source: https://github.com/sherlock-audit/2025-09-maple-sept-8th/issues/5

Summary

`GovernorTimelock::executeProposals()` calls the scheduled proposal target, but only deletes the proposal information after, allowing a malicious executor to reenter and execute the same proposal more than once.

Vulnerability Detail

As shown below, `GovernorTimelock::executeProposals()` doesn't follow the CEI pattern and allows a malicious executor to reenter in case the proposal passes execution to an executor controlled contract.

Impact

Depends on the use case, couldn't identify an attack path that resulted in significant damage.

Code Snippet

https://github.com/sherlock-audit/2025-09-maple-sept-8th/pull/1/files#diff-8976e5067f2102ea34580e0de5ec6762a73d214088306194672c326512646a77R173-R175

Tool Used

Manual Review

Recommendation

Delete the proposal and then make the call.

Discussion

lpetroulakis

The team has acknowledged and fixed the issue in https://github.com/maple-labs/globals-v2/commit/2b9f9b9ca0508b5f98d866b9319cb979e69836ab

Issue L-3: `GovernorTimelock::_call()` assembly revert block doesn't have the memory safe attribute

Source: https://github.com/sherlock-audit/2025-09-maple-sept-8th/issues/6

Summary

As per the Solidity docs, it's recommended to use memory safe.

> While we recommend to always respect Solidity's memory model, inline assembly allows you to use memory in an incompatible way. Therefore, moving stack variables to memory and additional memory optimizations are, by default, globally disabled in the presence of any inline assembly block that contains a memory operation or assigns to Solidity variables in memory.

This is also in line with the OZ implementation.

Vulnerability Detail

Essentially certain optimizations are disabled when memory safe is not set in the assembly blocks, which is not recommended.

Impact

Undefined but it's better to follow the recommendations.

Code Snippet

https://github.com/sherlock-audit/2025-09-maple-sept-8th/pull/1/files#diff-8976e5067f2102ea34580e0de5ec6762a73d2140883061946 72c326512646a77R225

Tool Used

Manual Review

Recommendation

Add memory safe as per the OZ implementation.

Discussion

lpetroulakis

The team has acknowledged and fixed the issue in https://github.com/maple-labs/globals-v2/commit/c19ed92f23af4903a0d2781f523394ea57936c13

Disclaimers