Security Audits with SlowMist and TrailOfBits

Background

On October 10th, 2018 we requested a Security Audit from SlowMist for the period of 1 week. Later, from Nov 5th to Nov 20th 2018 we requested a second audit from Trail of Bits. These two audits were done independently. The full final reports are each included in this repo.

The areas where we expected the greatest risk was with numerical stability (rounding errors or overflows in arithmetic) and upgrades.

The rest of this document will summarize the findings from each of these audits and detail Ampleforth Engineering's responses to each of them.

The git hashes representing Ampleforth's post-audit changes are:

uFragments
f0209fe71ee412e046b9e845cbd142869925a970

market-oracle
8e72da1b8d60b86b8aa968fb47176ba926e250c2

uFragments-security-tests
b8a12bebac2f08355b35ad25e4337286e0db9d1c

uFragments-eth-integration
ee46c18bd1c9f67201acdeb9a1be2ac7adac67d3

Frg-ethereum-runners
090d0589359a1466c7ad86c64a6a04bc3ef9ac18

SlowMist Response Summary

SlowMist found no vulnerabilities over the course of their audit. They made one recommendation of changing calls of assert() to require() to optimize gas usage. For a quick rundown on the difference, see here.

While it's true that assert uses more gas than require, that's only true in the case when the checked condition fails. We follow the practice of using require to verify input parameters and assert to verify system integrity in accordance with the documentation's recommendations linked above. Since assert should only fail when there is a serious bug in the system, gas efficiency isn't a concern for us there. Using assert also has the benefit of future support for automated code verification tools. For these reasons, we decided to keep the usages of assert intact.

Trail of Bits Response Summary

Overall, no High Severity or Medium Severity issues were found, and only 4 Low Severity were identified.

TOB-FRAG-001

Rebase will fail if no market sources are fresh

Previously, if no market sources were fresh the rebase operation would cause a revert with no change to the token supply. While the end result of this revert on the token supply was considered correct behavior, we did make two small code changes [#100, #34].

With these changes, both **no fresh sources** and **no trade volume** are considered valid conditions (hopefully ones we never encounter). Instead of rebase failing with no supply adjustment, rebase will now run to completion with no supply adjustment. This is a small conceptual change, but it does allow writing to the event logs that a null rebase occurred. In the process, we also added a configurable hyperparameter for minimum required volume before any supply adjustment is made.

TOB-FRAG-002

Malicious or erroneous MarketSource can break rebasing

A malicious market source, which has also gained whitelisted status, can cause an integer overflow in the calculation of the volume-weighted average in the oracle aggregator by supplying fake data with very large values. If this happens, rebase will fail with a revert and no adjustment to the token supply will be made. If a malicious market source continues to supply such a value, it can result in a denial-of-service attack on the monetary policy until it's removed from the source whitelist.

After some discussion, the Ampleforth team decided to take no immediate action. The fundamental issue is that the oracle relies on a whitelist of sources authorized to provide data---fixing an overflow with an input restriction still would not have changed this. Adding a maximum allowable value independent of the number of sources combined in the calculation would have either been arbitrary or overly limiting.

Truly decentralized oracles are the best approach long term, but they're still highly conceptual and not ready for a high stakes, adversarial environment. We're keeping a close eye on this space, and are considering migrating to external oracle infrastructure at some point, like Chainlink. It's worth noting that other prominent projects also use whitelisted sources, including for example MakerDAO and Compound.

TOB-FRAG-003

Zos-lib is deprecated

Ampleforth and Trail of Bits discovered this at roughly the same time. This library version was updated in [#98].

TOB-FRAG-004

Possible reentrancy if the minimum rebase interval is zero

If the minimum time between rebase events is set to zero, and particular malicious outside market source code were whitelisted in the oracle, then it would be possible for that source to reentrantly call rebase. The only observed downside in that execution scenario is that if an external observer relied on the monotonicity of epoch events, it would get them out of order.

We implemented Trail of Bits' recommendations in [#104], which requires the minimum rebase interval to be non-zero and the rebase timestamps to strictly increase. Long term we will consider having the market sources push data to the oracle aggregator rather than having the aggregator call into the market sources, or switch to alternative external oracle infrastructure. This is not specifically a response to the rentrancy issue, but a broader development aim of decentralizing the Oracle as much as possible over time.

TOB-FRAG-005

Marketsource removal is dangerous

The private helper function that removes market sources from the oracle relied on assumptions on the input, though all current usages of the function appear safe.

We considered adding explicit require checks in the code, but since it's a private function only callable internally we opted to save gas and document the input requirements instead. This was done in [#35].

TOB-FRAG-006

Contract upgrades can catastrophically fail if the storage layout changes

We rely on ZeppelinOS' library for upgradeability of the token and monetary policy contracts. This is to support ongoing maintenance and fix any unforeseen bugs. Eventually this upgradeability will either be removed or guarded via on-chain governance.

An issue that affects all projects that use ZeppelinOS' approach to upgrading is that each new version of the contract must maintain the same storage layout. There has been talk of adding a features to solc that emits the storage layout of contracts after compilation, but this feature does not yet exist.

In the interim, we'll carefully record the compiler version and configuration during deployment so we can maintain the same storage structure in future versions. The latest Truffle version, makes configuring compiler options very easy.
All the tooling needed for Migrations exist, but would only be considered in extreme cases. The overhead and integration challenges with other users and applications who interact with Amples makes migrations difficult.

TOB-FRAG-007
Rebase predictability may make μFragments a target for arbitrage
Ampleforth would like to encourage as much arbitrage as possible, actually! This helps the market converge on the true price.

It's important to keep two things in mind. First, that price is determined by the actors in the market, not by the protocol directly. And second, that prices do not instantaneously react after a rebase operation executes.

Market actors will always be able to view public markets and blockchain state, and price these data into their trades even in advance of rebases. We have a forthcoming dashboard that will make all the relevant onchain data available to every trader everywhere.

Property-Based Testing and Formal Verification
Trail of Bits also provided a custom Echidna testing harness for the ERC20 token and custom Manticore scripts. We have added them to sibling uFragments-security-tests repo. These serve to broaden our 100% unit test coverage and integration tests of the smart contracts with fuzz testing and symbolic execution tools.