TL;DR

Present to the community different security disclosures received during the last months, with our evaluation of each one of them, remediation, and ad-hoc proposal of bounties, required until a formal bug bounty program is defined.

Clarification on evaluation

At the moment, the Aave community isstill working on a bug bounty program, but even being a decentralized ecosystem, we think it is really important to fairly evaluate and reward bug reports of security researchers, as a key security mechanism of the protocol; even if this needs to be done ad-hoc like in this case.

There is currently a bug bounty program privately run by Aave Companies (not the DAO), with what we consider a bit outdated evaluation criteria and rewards, so even if used slightly as a reference, our criteria and ad-hoc proposals don't follow it.

Our evaluation and proposals on this post are based on the following, and will be extended to the more rigorous bug bounty program once set up:

- Objectivity, both on Severity/Impact and Likelihood

. We understand pretty well both aspects in regard to the Aave system and try to be completely fair and realistic concerning especially Likelihood.

Given that bugs are quite unique, both Severity and Likelihood ranges should be understood as that, ranges, with bugs belonging to the same ranges having slightly different rewards.

- Subjectivity on bounty amounts, based on the sustainability of rewards

. The Aave ecosystem is investing an important amount of funds continuously in all kinds of security procedures: audits, formal verification, and processes during the development itself. To keep the protocol sustainable, it is our duty to propose rewards (to be extended to the future bug bounty program) that we consider fair but sustainable.

Still, it is important to highlight that the approval of these amounts will be done by AAVE holders via Snapshot and an on-chain governance proposal.

Severity Ranges

Severity refers to the consequence to the Aave protocol, its users, and other entities of the ecosystem.

No impact

. There is a problem but without any practical or malicious implications.

Low

. Really mild impact, not disturbing operations of the contract, protocol, or users in a meaningful way.

Medium

. Impactful for the protocol and its users, but with clear and low limits due to the nature of the attack. Not involving potential loss of funds from users.

High

. Impactful and involving a potential loss of funds from users.

Critical

. Highest impact, usually putting at risk all the available liquidity in the protocol or a really meaningful percentage of it.

Likelihood ranges

In practice, the probability for the risk to materialize into an impact.

Not possible

. Not exploitable, but still wrong and requires a fix or action.

Not likely

. Theoretically possible, but not in practice for example due to the attack's complexity.

Possible, not likely

. Possible under specific circumstances, which could realistically happen.

Likely

. Possible under realistic circumstances.

Certain

. Sure, if not exploited/applicable in production, just by chance or a matter of time.

Reports

1. Flash loan premium not passed correctly to the receiver

On Aave v3, the fee model of flash loans has the following cases:

1. In the default scenario, the protocol itself charges a fee configured on the global Pool variable _flashLoanPremiumTotal

, which will get pulled from the receiver of the flash loan at the end of the action.

1. If the address initiating the flash loan is an authorized flash borrower, the fee applied is 0.
2. If instead of repaying the flash loan the initiator decides to keep a variable borrow debt position, the fee applied is also 0.

Even if calculations when pulling the flash loan fees were correct, the totalPremiums

input data sent to the receiver was not taking into account cases 2 and 3, potentially causing the flash loan receiver to approve the Pool to pull slightly more funds than required.

This has been fixed with https://github.com/aave/aave-v3-core/pull/832/files#diff-8e7cb86925395925a9f8d8fc1d0298ff3421b6985bad665e4ac502e1adafcc70R94 , included on the Aave v3.0.2 upgrade.

Reporter:

Emanuele Ricci (stermi)

Severity:

No impact

- With the Pool being a trusted entity by users of Aave with correct calculation of the amount pulled, the impact only affects potential optimizations of the flash loans' receiver, a consequence of maybe assuming that the totalPremiums

data input from executeOperation()

will indicate how much get pulled from itself.

Likelihood:

Certain

- This problem was present on the deployed versions, so even if not exploitable and minor, affects production.

Proposed bounty:

5'000 USD

2. Misusage of e-mode oracle feed after an asset is removed from e-mode

This problem resides on this logic branch https://github.com/aave/aave-v3-core/blob/master/contracts/protocol/libraries/logic/LiquidationLogic.sol#L425, used for returning the configuration data of the assets involved in a liquidation action.

The core of the problem is that the code assumes wrongly that a debt asset being liquidated should use an e-mode common oracle if:

1. The user liquidated has e-mode activated.
2. The collateral asset liquidated belongs that that e-mode.

3. The e-mode has a common oracle configured for all the assets belonging to it.

This is not true, because technically it is possible for some admin roles of the PoolConfigurator to for example remove the debt asset from e-mode, which would lead to directly using a wrong oracle source (the common on previous e-mode) instead of the correct one (the specific for the asset outside e-mode).

This problem will be fixed in a future upgrade of Aave v3.

Reporter:

Emanuele Ricci (stermi)

Severity:

High

- Wrong pricing of assets due to logic qualifies as High, as it could (and would) lead to wrong accounting on liquidation. However, this doesn't qualify as Critical because there can't really be a scenario in which a common e-mode oracle gets configured for multiple assets without them having an important price correlation when configured.

If for any reason that asset would get removed from e-mode, it should be because of a lack of price correlation, which by itself would create problems before this bug is anyhow exploited.

Likelihood:

Not possible

- e-mode common oracles have not been configured and there is no plan to activate them in the future, as the granularity provided by having a feed per asset even within e-mode is net better.

In addition, removing an asset from e-mode is a really critical step of even more magnitude than lowering the Liquidation Threshold, so there can't be a scenario in which any proposal doing it will be deeply evaluated and tested.

Proposed bounty:

10'000 USD

3. Griefing risk with LTV0 and isolated collateral assets

For both LTV0 and isolated collateral, the problem is the same, so this description will be focused on the first.

Aave v3 has a pretty powerful risk control mechanism on the so-called LTV0 feature.

In summary, whenever the protocol configures the LTV of an asset as 0, different from any other value on that parameter, this has implications on the current existing positions on Aave, as the protocol considers the LTV0 asset as with higher risk than others, so it should be off-boarded.

In practice, this forces the users to "remove" the LTV0 assets from their positions before they can remove any non-LTV0. For example, if a user has an LTV0 asset as collateral and a non-LTV0, in the average scenario he will need to withdraw first the LTV0 if he wants to withdraw the non-LTV0.

The reported problem is a consequence of an extra mechanism of Aave, which is that if a position receives aToken eligible as collateral, they get automatically enabled as collateral. This was including LTV0 assets too.

So the "griefing" appears in a situation like the following:

1. User A has asset X as collateral, a non-LTV0, and borrowings against it.
2. "Griefer" transfers him aToken of asset Y, which is enabled as collateral but LTV0.
3. Automatically, the asset Y will be enabled for user A as collateral
4. In order to withdraw non-LTV0 asset X, the user would need to withdraw Y first or disable it as collateral.

Even if not damaging, as the user can disable the asset as collateral before any action, this attack would hurt user experience, and in edge cases of immutable contracts, potentially create some temporary DOS on integrators.

This problem applies exactly the same way to isolated collaterals, as receiving a transfer of their aTokens will automatically enable them as collateral to the user; which will consequently create the same issues, with him forced to disable it as collateral.

This issue has been fixed on the Aave v3.0.2 upgrade.

Reporters:

LTV0:

Emanuele Ricci (stermi)

Isolated Collaterals:

cmichel

Severity:

Low

- Not taking into account pretty edge cases of certain immutable contracts, any exploit of the problem is reversible, mainly having consequences on the user experience.

Likelihood:

Possible, not likely

- An attacker should on purpose spend resources (e.g. gas) for an action that will be reversible by the victim. So even if possible especially in low-cost environments, there is not much rationale for it.

Proposed bounty:

20’000 USD each

4. Risk of price manipulation on GUNI USDC/UDST due to illiquidity

The bug report involves the low liquidity and limited number of holders (12 including Aave aToken) of G-UNI USDC/USDT, listed on the Aave v2 Ethereum AMM pool, which increases the risk of an oracle manipulation via the so-called “donation” attack.

The attack involves the following:

1. The attacker is able to reduce the number of holders of G-UNI USDC/USDT to only himself and/or the G-UNI USDC/USDT Aave aToken.

This is quite theoretical, as it would involve some type of bribing or social engineering on 11 holders at that point in time.

1. Flash loan of USDC or USDT.
2. Donation of the flashed funds to the G-UNI USDC/USDT pool.
3. Even if the attacker “loses” funds on step 3), the price of the G-UNI USDC/USDT (which he holds on Aave as collateral) gets highly inflated, which allows him to borrow all available liquidity on Aave at that point in time (approx. $2.8m)

Even if Low likelihood, and with the Aave v2 Ethereum AMM pool being off-boarded via freezing, assets with so low liquidity are not acceptable for Aave given their tail risk.

In addition to the existing barriers to setting up the attack scenario, the extra balance of G-UNI USDC/USDT has been further burnt, which makes impossible the attack.

Also, Aave v2 AMM was and is in the process of off-boarding.

Reporter:

kankodu (kankodu)

Severity:

Critical

- The impact of big price manipulations of assets listed on Aave followed by illicit borrowing of funds is by general rule critical if a pool is in a functional state.

Likelihood:

Not likely

- This attack depends not only on the so-called “donation”, but also on the attacker being the only entity holding the underlying assets (including or not the aAmmGUniUSDCUSDT aToken).

Given the coordination/involvement required with existing holders (11) at the moment of disclosure to proceed with a malicious exploit, the Likelihood of this happening is remote.

Proposed bounty:

20’000 USD

5. Inconsistent amount on aToken transfer events

This problem affects the event emission of aTokens (and debt tokens), more precisely the precision of the amounts emitted.

Due to how these tokens work internally with so-called scaled balances and indexes, all Transfer-related events should take them into account, to consider both “principal” and “yield accrued”, to be able to reconstruct precisely historic balances. The code was not precise on this, and it has been fixed HERE, included in the Aave v3.0.1 upgrade.

Reporter:

watchpug

Severity:

Low

- As it doesn’t affect on-chain infrastructure, the impact is low, but it exists regarding off-chain infrastructure.

Likelihood:

Certain

- Was present in production.

Proposed bounty:

10’000 USD

Contact

We tried to cover on this proposal all reports of which we are aware but given the decentralization involved in the reporting procedure, if any report is missing, contact us on Twitter or on hi@bgdlabs.com.

Only Aave smart contract reports

.