[

STK_banner2-colour

1920×845 116 KB

](https://europe1.discourse-cdn.com/business20/uploads/aave/original/2X/d/de54278ba6952ab4854798e440d610046c073995.jpeg)

TL;DR

We present to the community a technical upgrade of the Safety Module (SM) smart contracts (stkAAVE/stkABPT), facilitating slashing management and improving other aspects like cooldown mechanics.

https://github.com/bgd-labs/aave-stk-v1-5

Context

On September 2020, the Aave Safety Module was introduced into the ecosystem, to improve the protection of the liquidity protocol, adding an extra utility for the AAVE token: AAVE or AAVE/WETH BPT holders stake their assets to act as a defensive layer in front of any shortfall event.

Since then, apart from really minor upgrades, the contracts have remained the same and in parallel, the system has been adopted by other communities.

As part of our engagement with Aave, we identified that the Safety Module is a clear area of improvement in the ecosystem, from 2 different standpoints:

1. Technical

. Improving the basic existing mechanisms of the contracts, but without disrupting radically the current design of the SM.

1. Conceptual

. Making more efficient the SM dynamics (e.g. slashing, rewards distribution rules, etc), by modifying the whole design.

The conceptual improvements should be discussed separately by the community, as they belong more to the field of AAVE token dynamics. In this document, we outline the Technical upgrade.

Safety Module 1.5

Considering the currently deployed version of the Safety Module as v1, the one we present to the community can be addressed as v1.5, being more of a technical improvement, and not a complete revamp of the system that would qualify as v2.

In this document, we will be referring to stkAAVE, but everything described will also apply to stkABPT.

1. New slashing mechanism

On the running SM v1, in order to slash, an ad-hoc governance proposal is required, involving important development overhead, which is not ideal.

The new SM v1.5 adds an enhanced mechanism to facilitate slashing of the underlying by tracking an exchange rate between the staked AAVE and the stkAAVE received by stakers. The mechanism is simple: when somebody stakes AAVE, they receive a certain amount of stkAAVE, which no longer is 1:1 equivalent, as it keeps track of slashing (meaning with the same stkAAVE, stakers are able to claim less AAVE).

Given that in a slashing scenario is highly probable that the slashed amount with not be fully used to cover a shortfall event, the exchange rate introduced also supports any entity to “return” or “donate” funds to stakers, necessary in the case of over-slashing to account for price volatility on a potential shortfall event.

A full slashing flow could be as follow:

1. The protocol suffers a shortfall of 100’000 DAI, which should be covered by slashing AAVE from the SM.
2. The slashing admin (explained later) slashes 2000 AAVE, equivalent to ~160’000 DAI, to account for a potential price depreciation of AAVE during the time to cover the shortfall. The exchange rate gets updated, as now stkAAVE has fewer claims over staked AAVE.
3. By some auction mechanism (out of the scope of the SM at the moment), finally only 1500 AAVE are required to cover

the 100’000 DAI shortfall, so 500 AAVE can be returned back to stkAAVE stakers.

4. The system holding the 500 AAVE (e.g. the Aave Collector, or any escrow contract of an auction) interacts with the SM to return the funds to stakers. The exchange rate is updated, as now stkAAVE has recovered some extra claims over staked AAVE.

5. The slashing admin marks the slashing action as settled, and the Safety Module returns to its normal functioning.

From the previous flow, it is important to highlight certain new dynamics and roles:

- Slashing admin

. The only entity with permission to trigger the slash()

function on the Safety Module contract. Taking into account the time constraints required for an efficient slashing (should be fast enough), and also considering that governance is not fundamentally affected by the action, initially to be assigned to the Level 1 (Short) Executor

.

Additionally, the slashing admin is in charge of defining the maximum slashing percentage that can be slashed at once, currently determined by the community as 30%.

- Running vs Post-Slashing states

. Whenever a slashing happens, by definition the Safety Module enters a slightly unbalanced state, as potentially some (or even all) funds could be returned.

In order to not allow any kind of arbitrage, after slash and until the slashing admin decides to settleSlashing()

, the system enters a Post-Slashing state

, during which:

- Staking is disabled

. Given that is unpredictable if funds are gonna be returned or not from slashing “leftovers” no extra staking is allowed.

- Cooldown is disabled

. If anybody wants to redeem their assets, no cooldown needs to be respected, as it is reasonable to give a choice to stakers to evaluate if the risk/reward of staying is worth it. Important to clarify that staking rewards keep accruing, so the “pressure” of stakers redeeming will accentuate the rewards of those staying.

- No extra slashing can be done

. In order to respect the maximum slash percentage, it is not possible to slash without settling first the current slashing.

The Running state

is the non-Post-Slashing, so just the “normal” situation of the SM.

- Staking is disabled

. Given that is unpredictable if funds are gonna be returned or not from slashing “leftovers” no extra staking is allowed.

- Cooldown is disabled

. If anybody wants to redeem their assets, no cooldown needs to be respected, as it is reasonable to give a choice to stakers to evaluate if the risk/reward of staying is worth it. Important to clarify that staking rewards keep accruing, so the “pressure” of stakers redeeming will accentuate the rewards of those staying.

- No extra slashing can be done

. In order to respect the maximum slash percentage, it is not possible to slash without settling first the current slashing.

2. New cooldown mechanics

The current cooldown on SM v1 consists of a time delay of 10 days to be respected whenever anybody wants to redeem the staked AAVE. At the moment, this cooldown is affected by in/outflows of stkAAVE, both via transfer()

and stake()

/redeem()

, in order to protect the system from being gamed.

This mechanism is not really optimal and adds important complexity, so on v1.5 has been changed to the following: after activation of cooldown, a staker will be able to redeem the minimum balance he will hold between cooldown activation and redeem window

.

Example

- Staker A has 100 stkAAVE, and activates cooldown.
- After 2 days (so with the current 10 days of cooldown, 8 days left) Staker B sends 200 stkAAVE to Staker A, making his balance 300 stkAAVE.
- The cooldown period of Staker A remains unaffected (8 days), but he will only be able to redeem the initial 100 stkAAVE he has.
- In order to redeem the full current 300 stkAAVE balance, Staker A can activate again the cooldown, but this will reset the delay to the initial 10 days.

To avoid complexity, if the staker activates the cooldown and decides to transfer out part of his stkAAVE, that outflow amount will be discounted from what he will be able to redeem once the delay finishes.

Apart from the user-level cooldown new mechanics, we also propose to increase the cooldown period from 10 days to 20 days

.

The rationale is that the Level 1 (Short) governance procedure has a duration of approximately 5 days, so having a 4x longer cooldown period will act as an important deterrent for anybody trying to game the system by continuously activating cooldown.

3. Aave Governance Implications

Even if stkAAVE has voting power on the Aave governance, technically that power belongs to the AAVE claims by stkAAVE holders, which at the moment are 1:1 equivalent.

As with v1.5 the 1:1 equivalence will potentially not be respected anymore, this upgrade will also adapt the Aave governance strategy to take into account the exchange of stkAAVE with AAVE, in order to not under/overcount the stkAAVE influence.

4. GHO

The Aave community has already approved the deployment and activation of the GHO stablecoin and the first facilitator will be the Aave v3 Ethereum pool.

In order to enable the discount mechanism by holding stkAAVE described on the GHO proposal, the design of Aave Companies requires to introduce of a piece of logic on the stkAAVE transfer(), in order to “notify” the GHO facilitator system of stakers’ balances.

We have evaluated this and we think it is acceptable, so it will be included in this upgrade. From a technical perspective, stkAAVE includes logic protections to remain unaffected if anything would go wrong with the GHO facilitator, which at the same time should be considered a trustable entity, as it will be controlled by the Aave governance.

5. Misc

- The new version contains a stakeWithPermit() function to facilitate staking via signature by relayers or intermediate contracts, possible on stkAAVE given the underlying permit() of the AAVE token.
- The contract includes new “convenience” functions like claimRewardsOnBehalf()

, claimRewardsAndRedeem()

, or claimRewardsAndRedeemOnBehalf()

, permissioned to an entity “claim helper”, not included in this scope but that can be done in the future.

- On a technical note, the new implementation follows a preview*()

approach for stake and redeem actions, following similar standards as 4626, even if it is not the current objective to be fully compliant.

- The contracts can potentially be used by other communities/projects looking for the same functionality, but it is important to keep in mind that the current implementation is designed taking into account the existing production deployment, so customization (e.g. using completely different tokens to AAVE) should be done carefully.

Security

Security procedures by both providers of the Aave DAO (SigmaPrime and Certora) are getting finished at the moment, with an important emphasis on formal verification via Certora properties.

Additionally, we have followed our standard internal practices, including testing, code, and storage layout comparison with the previous implementation.

Finally, we appreciate the peer review done by @miguelmtz and @Zer0dot from Aave Companies.

All the information can be found in the Security section of the repository.

Next steps

Given this is an important development, we will be creating a Snapshot vote in the following days, and if approved by the community, once the security procedures are completely finished, we will submit the Aave governance proposal to execute the upgrade of the system on Ethereum.