Aave V3 Technical Paper

Emilio Frangella
emilio@aave.com

Lasse Herskind
lasse@aave.com

January 27, 2022

1 Introduction

Since its launch in 2020, the Aave Protocol (or "Aave") has driven innovation in the DeFi ecosystem. Controlled by Aave governance and with an IPFS-deployed and open-sourced frontend, Aave remains one of the biggest DeFi protocols, reaching peak liquidity of $30 billion. Key features of Aave (e.g., aTokens, choice of stable or variable interest rates, credit delegation, and more) have become industry standard, and remain a foundational layer of the protocol. But, as is true with all technology, the development of the ecosystem, along with continuous research and data analysis on markets as well as community feedback, have highlighted some areas for improvement of the current iteration of the Aave protocol.

These enhancements concern four specific areas:

1. Capital efficiency
2. Protocol safety
3. Decentralization
4. User experience

The design of Aave V3 implements enhancements in the categories mentioned above, improving user experience while offering increased capital efficiency without sacrificing security.

Before introducing the new features of V3, this next section digs deeper into why these four areas are ripe for improvement.

1.1 Capital Efficiency

Generate more yield for liquidity providers The total liquidity of the Aave Protocol across multiple networks is close to $20 billion. Most of that liquidity currently sits idle in the protocol smart contracts, generating yield from the borrowing activity. Although this yield is consistent and secure, it could be increased by implementing new user-facing functionalities that reuse the idling capital without increasing the solvency contingencies. (Note that this does not include reallocating the assets to other protocols for "yield farming" – this usually brings considerable smart contract risk and does not fit with Aave being a layer 0 DeFi protocol – where other protocols often allocate their assets to boost their yield.)

Optimize borrowing power In the previous iterations of the Aave Protocol, borrowers face challenges in maximizing their borrowing power in certain situations due to the fact that within the pool model, any collateral can be used to borrow any "borrowable" asset. Accordingly, the risk parameters for borrow transactions are set to be quite conservative.

Inefficient underlying network Most capital supplied to the Aave Protocol is on the Ethereum L1 network, which is saturated and has resulted in high transaction fees and UX pain points for users. The network inefficiency also reflects on the oracles' ability to submit accurate prices with a short response time.

Aggregated liquidity approach In the Aave Protocol, the total collateral value of a user is calculated by aggregating the value of all collateral deposited and normalizing the total to a certain base currency (usually ETH). The total collateral value is then used to calculate the average borrowing power, calculated as the weighted average of the borrowing power of all the individual assets (see the Aave whitepaper[1], section 5 for more details).The risk configuration of the assets reflects the fact that both the total collateral value and the total value borrowed can be volatile, and thus, are set conservatively. This limits the power borrowers can achieve from their collateral in many cases. For example, a user borrowing stablecoins while supplying stablecoins would have relative low volatility between collateral and assets borrowed and should therefore enjoy higher collateralization power.

Reduce liquidity segregation New liquidity protocols have sought to improve collateralization power while reducing risk by enabling either isolated pools or isolated pairs. Although this could improve collateralization power for certain assets, this actually encourages additional liquidity segregation (liquidity providers are forced to deploy assets across multiple pairs/pools to match their risk appetite) and UX issues (borrowers may be forced to allocate their collaterals to different pools/pairs to be able to borrow what they need). Section 2 analyzes in greater detail some of the current approaches and areas for improvement.

New liquidity protocols have sought to improve collateralization power while reducing risk by enabling either isolated pools or isolated pairs. Although this could improve collateralization power for certain assets, this actually encourages additional liquidity segregation (liquidity providers are forced to deploy assets across multiple pairs/pools to match their risk appetite) and UX issues (borrowers may be forced to allocate their collaterals to different pools/pairs to be able to borrow what they need) analyzes the current approaches and their drawbacks more in detail.

1.2 Protocol safety

Risk management is crucial for a liquidity protocol, requiring mitigation of smart contract risk and liquidity risk. For smart contract risk, a careful review and

[1] https://github.com/aave/protocol-v2/blob/ac8897348c1abc4a74659af254ade8900007faba/aave-v2-whitepaper.pdf

audit process of any code update and governance proposal is needed. This is mostly an off-chain/coordination matter, and managing this risk usually involves maintaining a mission-critical culture when handling code/smart contract upgrades. The liquidity risk is more subtle, involving careful evaluation of the market conditions to tune the parameter configuration and proper due diligence on assets listing proposals by the community (the Aave community developed and maintains the Aave risk framework[2]).

Risk configuration in the current iteration of the Aave Protocol is unfortunately limited. While it is possible to adjust borrowing power (LTV) and maintenance margin (liquidation threshold) on the fly, the protocol could benefit from increased defense measures against possible strikes such as infinite minting or oracle manipulations.

1.3 Decentralization

Protocol management is completely decentralized and thus, controlled by AAVE token holders ("Aave governance"). Aave governance acts as a gatekeeper in certain aspects of the protocol configuration (e.g., listing new assets). Although this is important to ensure the safety of the protocol, this is somewhat limiting for teams and projects looking to have access to the Aave Protocol's liquidity by listing a token.

1.4 User experience

In a multichain and multi-rollup world, allowing liquidity to flow seamlessly across different chains is increasingly important. While the Aave Protocol is currently deployed on numerous networks with meaningful TVL, there is no seamless way for users to move liquidity from an Aave instance on one network to an Aave instance on another network.

2 Current solutions

Certain recent liquidity protocols implemented alternative designs to try and address these issues – most notably, isolated liquidity pairs and isolated liquidity pools. Both of these solutions seek to address the permissionless aspect of listing new assets while at the same time reducing risk of insolvency. Although both design features are beneficial, they also highlighted ways that Aave V3 could further enhance and improve upon these models.

2.1 Isolated liquidity pairs

This approach uses what is conventionally defined as "isolated pairs" – supply/collateral pairs in which borrowers can only borrow one asset at time with a specific collateral. This has some advantages compared to the aggregated pool approach of Aave V2:

Reduced gas costs In general, the gas footprint of the interactions is reduced as there is less state management and no need to iterate across all the assets the user has borrowed and supplied.

[2] https://docs.aave.com/risk/

Permissionless listing Any asset can be listed with very limited risk management.

This design, however, also influences the ability of protocol's ability to scale.

Extreme liquidity segregation Liquidity is now strictly segregated per collateral, meaning that suppliers might be forced to split their capital across multiple pools to chase the highest yields. Though supplied cryptoassets could be aggregated through tools built on top of the protocol, this reduces users' ability to fine-tune their risk profile, partially invalidating at least one of the main advantages of this approach. This also means that for each new pair, liquidity needs to be built for borrowers to be able to borrow against their newly listed asset.

Bad UX for borrowers In the Aave Protocol, the act of borrowing funds is straighforward as both the supplied collateral and the borrowed crytpoassets are aggregated. This offers a very simple UX: borrowers can instantly borrow any asset as long as the user supplies the required overcollateralization, and only one position needs to be managed. With isolated pairs, borrowers must interact with many pairs at the same time if they want to borrow multiple assets, even if a single form of collateral is used. The consequence of this is that many positions must be managed simultaneously. This can be mitigated by building solutions on top of the protocol, but it reduces the impact of the liquidity segregation and also invalidates one of the main advantages (gas costs). Liquidity segregation also means in general higher average borrowing rates.

Skewed towards more risky assets In general, borrowers (on other liquidity protocols) who use risky assets as collateral are willing to pay more compared to safer assets. Such user behavior may allow for potential attacks that are possible in protocols that allow permissionless asset listing without any sort of debt ceiling. For example, users might improperly supply stablecoins to a newly created pool with a volatile or otherwise unstable asset by artificially borrowing against that asset, which increases the attractiveness for suppliers.

2.2 Isolated liquidity pools

Isolated pools define an architecture that is essentially a hybrid between Aave and isolated pairs, e.g., a permissionless protocol to instantiate indipendent markets with specific assets and custom risk management configurations. The approach is similar to what Aave pioneered in 2020 with the first Uniswap pool to borrow against Uniswap V1 LP assets. Having multiple markets enables users to split the risk across cluster of assets, reducing the impact on potential asset failures without sacrificing on UX and some compromises on liquidity segregation.

Given its specific architecture, this approach suffers from some of the issues as isolated pairs. Although not as severe, liquidity segregation is still important – many pools struggle to attract liquidity. TVL also is generally skewed towards more risky assets

3 Aave V3 Overview

The Aave V3 design is born from thoughtful analysis of the evolution of the protocol and its ecosystem's. The V3 enhanced features allow for new use cases that will spur a wave of innovation from users and developers. Aave V3 makes improvements in all the aforementioned categories – capital efficiency, security, decentralization, UX – while simultaneously providing new functionalities to leverage the capabilities of rollups and the growing ecosystem of competing L1s.

3.1 Capital efficiency and UX improvements

Portal Portal represents a new set of core features that can be used to allow supplied assets to seamlessly flow between Aave markets on different networks. On a high level, the feature is very simple: the protocol leverages the unique pegged design of the aTokens to burn aTokens on the source network while minting them on the destination network. This enables a way to supply assets in a deferred way, where the underlying cryptoasset is supplied to the Aave Protocol after it has passed through the canonical chain bridge. Further details on the design can be seen in section 4.5.

E-Mode High Efficiency Mode (E-Mode) allows borrowers to extract the highest borrowing power out of their collateral. Assets can now be categorized, with each category having the following risk management parameters:

- LTV (Loan-to-value)
- Liquidation threshold
- Liquidation bonus
- A custom price oracle (optional)

E-Mode allows borrowers to restrict themselves in borrowing only assets belonging to a certain category (e.g., stablecoins). When this happens, if users supply assets of the same category as collateral, the borrowing power (LTV), and maintenance margin (liquidation threshold) are overridden by the E-Mode category configuration to allow higher capital efficiency. See fig. 1 for an illustration with multiple categories.

With V3, the Aave Protocol can support a maximum of 255 E-Mode categories, enabling a wave of new use cases such as:

- Highly efficient yield farming (for example, deposit ETH staking derivatives to borrow ETH)
- Diversified risk management

Example The Aave Protocol defines E-Mode category 1 (stablecoins) as: 97% LTV, 98% liquidation threshold, and 2% liquidation bonus, without a custom price oracle.

1. User chooses E-Mode category 1 (stablecoins)

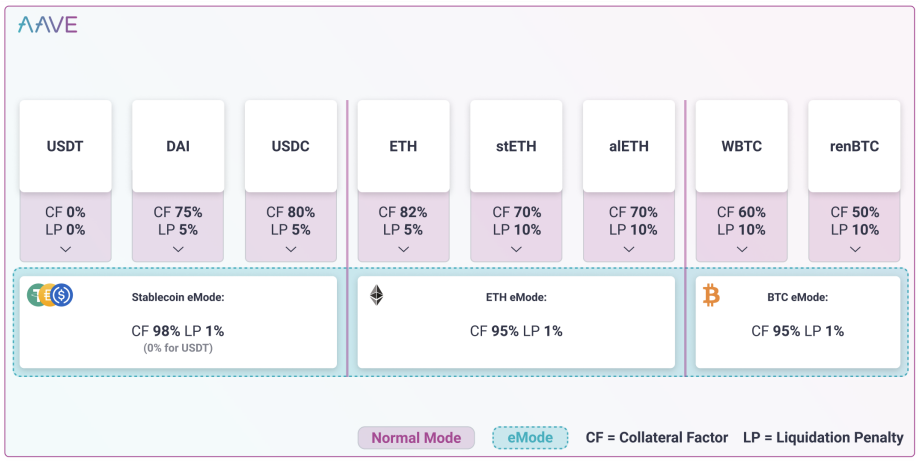

Figure 1: Illustration with three E-Mode categories (Stables, Eth, BTC)

2. User supplies DAI (which normally has 75% LTV)
3. User is now allowed to borrow other stablecoins (including DAI) with the borrowing power defined by the E-Mode category (97%). User's capital is therefore 22% more efficient. Note that the user is still allowed to supply other assets and to use them as collateral, but only collateral belonging to the same E-Mode category chosen by the user will have enhanced risk parameters.

Isolation Mode One of the most relevant areas for enhancement is navigating the inherent complexity of modulating risk exposure. When Aave governance lists an asset on the Aave Protocol, borrowers using it as collateral immediately have access to the whole protocol's liquidity. This makes it complicated to list new assets and reduces the capital efficiency for borrowers. Therefore, V3 introduces Isolation Mode, inspired by the MakerDAO approach for exposure management.

Assets can now be listed (if voted on by Aave governance) as "isolated". Borrowers supplying an isolated asset as collateral cannot supply other assets as collateral (though they can still supply to capture yield). Only stablecoins that have been permitted by Aave governance to be borrowable in isolation mode can be borrowed by users utilizing an isolated collateral up to a specified debt ceiling.

In Figure 2, the user is supplying $TOKEN2 as collateral. $TOKEN2 is an isolated asset with a maximum debt ceiling of $10M. The user will therefore be allowed to borrow up to $10M of USDT, DAI and USDC (the only three assets allowed to be borrowed, in this example, in isolation mode) and will not be allowed to activate any other asset as collateral (user can still supply ETH, for example, for yield generation). Users can exit isolation mode at any time by disabling $TOKEN2 as collateral (as long as it doesn't cause liquidation). $TOKEN2 can exit isolation mode when the Aave governance decides to remove the debt ceiling. This allows for a true permissionless listing in the future with exposure management.

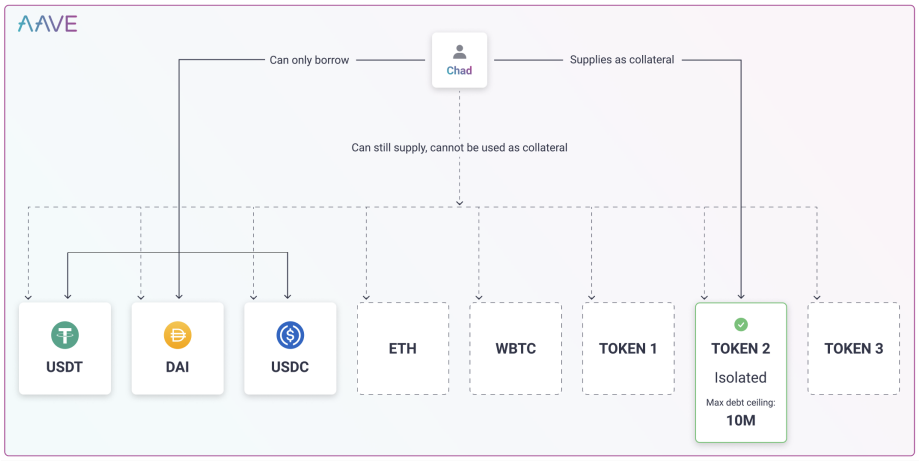

Figure 2: Illustration of user supplying $Token2 as collateral and borrowing stablecoins.

3.2 Risk Management

Aave V3 brings more sophisticated risk parameters and features to provide to the protocol a high level of protection against potential insolvency.

Supply and Borrow Caps Aave governance can now configure Borrow Caps and Supply Caps. Borrow Caps, similar to what is implemented in other liquidity protocols, allow the protocol to modulate how much of each asset can be borrowed. Supply Caps allow the limitation of how much of a certain asset can be supplied to the Aave Protocol. This helps in reducing exposure to a certain asset and mitigates risks such as infinite minting or price oracle manipulation.

Granular borrowing power control Currently, various liquidity protocols do not allow for reduction of the borrowing power of an asset without eventually causing liquidations. This is particularly limiting when the risk profile of a certain asset changes – e.g., it becomes impossible to reduce the exposure to that asset without liquidating previous borrowers. With granular borrowing power control in Aave V3, Aave governance can lower the borrowing power of any asset to as low as 0% without impacting existing borrowers (though it is still possible to liquidate existing users if deemed necessary).

Risk admins Aave V3 introduces the ability for the Aave governance to put entities on a permitlist to unlock the ability to change risk parameters without needing a governance vote. These entities can be DAOs (e.g., RiskDAO) or automated agents that can build on top of this feature to react automatically if certain invariants are broken.

Price Oracle sentinel The Sentinel feature has been designed for L2s to handle eventual downtime of the sequencer (but can be extended to handle other cases, even on L1s, in the future). It introduces a grace period for liquidations and disables borrowing under specific circumstances.

Variable liquidation close factor In Aave V2, liquidations can only liquidate half of the position at any time. This has shown to be inefficient in many cases, especially with high transaction fees and/or small positions. In V3, this mechanism has been improved to allow the position to be fully liquidated when it approaches insolvency (HF < 0.95).

3.3 Decentralization

Aave V3 introduces Asset Listing Admins. The Listing Admin is a specific role that can be granted by Aave governance to allow different asset listing strategies other than an on-chain vote as seen in Aave V2. This will allow builders to create custom asset listing strategies which can be designed to bring true permissionless asset listing.

3.4 Other features

- All the functions that involve transfer of tokens (supply, repay) now support EIP 2612 permit (this is especially important for L2).
- EIP 712 signature for credit delegation (no need for contracts to request a user transaction anymore).
- Repay with aTokens: Allows borrowers to repay using aTokens instead of the underlying.
- Permitlist on flashloans: Governance can permit entities to access free instant liquidity (waives flashloan premium).
- Protocol fee on liquidation (configurable).
- Protocol fee on instant liquidity (configurable).
- Simplified flashloans: A new `flashloanSimple()` allows to reduce impact on gas consumption up to 20% compared to the standard instant liquidity function (still available).
- Price oracle logic reworked to generalize calculations on the base asset (no longer ETH only).
- Gas optimizations: Despite all the new features, gas cost of all the functions dropped by around $20 - 25\%$ across the board.
- Code reorganized to be more modular.
- Smart contract re-engineering has greatly reduced the code size (more margin for other changes in the future) $\rightarrow$ up to 100K optimizer runs!
- New interest rate strategy that optimizes stable rate calculations (no more oracles for average stable borrow rates). The implementation follows the community discussion here https://governance.aave.com/t/base-stable-rate-oracle-update-and-improvements-in-aave-v2/1879, for further information see subsection 4.4.

4 Feature specification

4.1 Efficiency Mode

The High Efficiency mode feature is designed to maximize capital efficiency when collateral and borrowed assets are correlated in price, particularly when both are derivatives of the same underlying asset. Stablecoins are usually pegged to a particular underlying assets (for example, one USD), and the occurrence of dramatically losing the peg is rare. ETH and its derivatives (stETH, sETH, alETH and so on) behave in a similar way as their underlying (ETH). In these cases, there is the possibility of granting very high collateralization power.

In the original design of the Aave Protocol, the collateral value and the borrowed assets are usually normalized to a base currency (ETH or USD) and there is no way of knowing on-chain which collateral is backing what borrowed asset, which makes it very hard to improve collateral efficiency.

E-Mode introduces an asset classification that places all the assets listed on the Aave Protocol in a specific category; assets of the same category are usually highly correlated in price. The correct categorization cannot be enforced on-chain and needs to be maintained by the entity managing the protocol (typically the Aave governance).

Users can deliberately choose to borrow one specific category, and can therefore employ high collateralization power when using collateral belonging to the same category.

E-Mode also offers the possibility of introducing a specific price oracle for a certain category. For example, for a category where only renBTC and WBTC are defined, it would be possible to use the BTC/USD oracle for both of them whenever the user is in BTC E-Mode. This would further reduce the risk of unwanted liquidations as it eliminates the oracle asynchronicity (in case of dramatic price drops of BTC, the WBTC/USD and renBTC/USD oracles might update at slightly different times since they are asynchronous; this might cause unwanted liquidations). The category-specific oracle introduces more risk for the protocol, as if one of the assets fails to keep its value (because of a protocol hack or an underlying issue of the specific derivative, for example) it might cause insolvency. The Aave governance will need to carefully evaluate on a per-asset and per-category basis whether or not to use the category-specific oracle. For all the reasons above, E-Mode is more suitable for faster networks (especially rollups) where the oracles can be more efficient and liquidations do not incur due to high transaction costs.

Invariants

Category 0 Category 0 is reserved as the default, non-E-Mode category. All the assets listed have this category by default, which indicates the standard operational mode.

Enter E-Mode A user can enter E-Mode (category $\neq 0$) only if **all** borrowed assets are in the chosen category.

Exit E-Mode A user can exit E-Mode (set category to 0) only if leaving E-Mode doesn't leave their position undercollateralized (HF ≥ 1).

User borrowing A user in E-Mode can **only** borrow assets of the chosen category. The user may use any asset as collateral, but only assets of the same E-Mode category will enjoy the higher collateralization factors.

Asset addition An asset can be added to E-Mode by authorized entities (risk or pool admin) only if the LTV and Liquidation Threshold of the E-Mode category is higher than the default, non-E-Mode risk parameters for the asset.

Asset removal An asset can be removed from E-Mode by authorized entities (risk or pool admin). This may drop solvency of some users into unwanted liquidations (see section 4.7). By design, users borrowing an asset in E-Mode that has subsequently been removed from an E-Mode category are unaffected. Users that are still in E-Mode will not be able to borrow that asset anymore after the removal.

4.2 Isolation Mode

In the current generation of liquidity protocols, including Aave V2, when a new asset is listed, the whole liquidity pool is exposed to it, meaning that users using the new collateral asset might be able to borrow the whole liquidity available. This greatly limits the capability to add new assets to the Aave Protocol, as every new asset increases the liquidity and solvency risk. Isolation mode solves for this.

Isolation mode allows the protocol to list assets as isolated; isolated assets have a specific debt ceiling (representing the maximum amount in USD that can be borrowed against the collateral, with two decimal points of precision). Borrowers using an isolated asset as collateral can only use that particular asset as collateral, and cannot enable any other assets (including other isolated ones). Users using isolated assets can still supply other assets for yield generation.

Assets that can be borrowed in isolation mode are indicated by the flag `BORROWABLE_IN_ISOLATION` and should be in USD stablecoins, although the design allows to choose any denomination for assets borrowable in isolation. The assets that are borrowable in isolation mode need to be of the same family, otherwise the debt ceiling calculations would be inconsistent; this consistency cannot be enforced at the smart contract level and governors should be careful when allowing an asset to be borrowable in isolation.

Invariants

Isolated assets Any asset with > 0 debt ceiling is an isolated asset

Adding isolated asset An asset can only be configured as isolated when there is no supplied liquidity (enforced by `_checkNoSuppliers()`)

Isolated collateral Users supplying an isolated asset as collateral will only be allowed to use that asset as collateral; in no other case (supply, transfer, liquidation, set as collateral) should it be possible to enable a non-isolated asset as collateral.

Supplying isolated Users supplying other assets and using them as collateral can still supply isolated assets for yield generation. It should never be possible

to enable the isolated asset as collateral when the user has already supplied other nonisolated assets and has these assets enabled as collateral.

Exit isolation mode Users can exit isolation mode by disabling the isolated asset as collateral. Given the nature of isolated assets, this can only happen if the user is not borrowing.

Removing asset from isolation mode Authorized entities can remove an asset from isolation mode at any time.

Debt ceiling Users should be forbidden to borrow if the isolated debt ceiling is reached (debt ceiling does not include interest accrued over time, only the principal borrowed).

4.3 Granular borrowing power control

The granular borrowing power control design consists of splitting the usual collateral factor used in most liquidity protocols in LTV (which defines the borrowing power for new borrows) and liquidation threshold (which defines the maintenance margin or, in other words, the collateral/debt ratio at which the position is considered undercollateralized and therefore subjected to liquidation). This capability of setting the borrowing power independently from the maintenance margin allows to more granularly control the risk associated with a specific asset while avoiding impacting existing users. In Aave V2, this feature was already implemented, but it was mostly considered a soft protection against borrowers borrowing up to the maintenance margin and thus being immediately liquidated.

Example Alice wants to borrow with an asset that has 0 LTV and the liquidation threshold > 0. It should be impossible, but Alice can:

1. deposit an asset with $LTV > 0$ (for example, using flashloans)
2. supply the asset with $LTV == 0$
3. borrow
4. withdraw the asset with $LTV > 0$

This would leave the protocol with a position where the collateral had in origin $LTV = 0$.

Aave V3 introduces a stricter rule on LTV, so it is now possible to enforce actual 0 borrowing power while preventing the situation explained above. This protection requires that borrowers using multiple assets as collateral – one of more of which have $LTV == 0$, are required to withdraw these assets first and thus cannot withdraw asset with $LTV > 0$. In the example above, in V3, Alice would be be permitted to execute the last step. Alice would be required to withdraw the asset with $LTV == 0$ first. V3 therefore prevents the situation entirely. In general the granular borrowing power control enforces that:

- Users can borrow against any asset as long as $LTV > 0$ and liq threshold > 0

- If an LTV of an asset is reset to 0, user should not be allowed to borrow against that asset anymore.
- Borrowers using mutliple assets as collateral who wish to withdraw must withdraw all 0 LTV assets before any other assets an be withdrawn. This is true for withdrawal as well as transfer. Liquidations, though, are still allowed against assets that do not have 0 LTV.
- It is still acceptable for borrowers who want to boost their borrowing power (getting closer to liquidation threshold) to use the procedure explained in the example above. The delta LTV $\leftrightarrow$ liquidation threshold represents a soft protection for borrowers and a way to reduce liquidation risk on average, but the liquidation threshold is still considered safe for the protocol.

4.4 Stable interest rate oracle removal

The new interest rate strategy implements an algorithmic way of managing stable rates, as discussed on the governance forum[3].

The current implementation drops the lending rate oracle in favor of optimizing the stable rate against a certain optimal stable/variable debt ratio, defined by the interest rate strategy[4] and sets the minimal stable debt APR as $slope_{v,1} + \text{offset}_{base}$.

As an example, for an asset with a stable offset of 2% that is hitting 4% variable rates at its optimal utilization of 90%, the minimal stable rate will be 6%. The rate is then computed using the slopes $slope_{s,1}$ and $slope_{s,2}$ and further offset with at most offset_{excess} if the stable/variable debt move beyond the optimal.

To compute the stable interest rate, let O_{util} and O_{ratio} be the optimal utilization rate and optimal stable to total debt ratio constants respectively, then compute the stable interest rate as seen below:

$$base_s = slope_{v,1} + \text{offset}_{base}, \qquad E_{util} = 10^{27} - O_{util} \tag{1}$$

$$ratio = \frac{debt_{stable}}{debt_{stable} + debt_{variable}} \tag{2}$$

$$rate_s = \begin{cases} base_s + slope_{s,1} + slope_{s,2} * \frac{(util - O_{util})}{E_{util}}, & \text{if } util > O_{util} \\ base_s + slope_{s,1} * \frac{util}{O_{util}}, & \text{otherwise} \end{cases} \tag{3}$$

$$rate_s = \begin{cases} rate_s + \text{offset}_{excess} * \frac{ratio - O_{ratio}}{10^{27} - O_{ratio}}, & \text{if } ratio > O_{ratio} \\ rate_s, & \text{otherwise} \end{cases} \tag{4}$$

4.5 Portals

Portal represents a set of core features that can be used to allow supplied assets to seamlessly flow between Aave instances on different networks. On a high level, the feature itself is very simple:

[3] https://governance.aave.com/t/base-stable-rate-oracle-update-and-improvements-in-aave-v2/1879

[4] OPTIMAL_STABLE_TO_TOTAL_DEBT_RATIO

The Aave Protocol leverages the unique design of the aTokens to burn aTokens on the source network while instantly minting them on the destination network. The underlying assets can then be supplied to Aave on the destination network in a deferred manner, by passing it to the pool *after* it has been moved through a bridge. This has a number of implications on how the interest rates are calculated and in general the security of the market on the destination network.

Design-wise, very little functionality is required at the protocol-level to support multiple very different portal implementations. Only three additional features are required by the protocol: *i*) minting unbacked aTokens, *ii*) backing the unbacked aTokens and *iii*) a whitelisting mechanism for contracts that want to use those features. Besides these features, additional internal accounting of the unbacked amount per-reserve and an update to the interest rates computation was needed.

Note that minting of unbacked aTokens does not influence the utilization rate for borrowers as no additional liquidity is provided ($util_{V2} = util_{borrow}$), however, it *must* be accounted for in the supply utilization because the unbacked aTokens are accruing interest. To accommodate such, we compute separate utilization rates for the borrow and supply side as fallows:

$$debt_{total} = debt_{variable} + debt_{stable}, \qquad total\_liq = debt_{total} + avail \tag{5}$$

$$util_{borrow} = \frac{debt_{total}}{total\_liq}, \qquad util_{supply} = \frac{debt_{total}}{total\_liq + unbacked} \tag{6}$$

As visible in Equation 6, increasing the *unbacked* value will lower the supply utilization and thereby the interest earned by the liquidity providers because they are diluted by the minted tokens. To offset this interest dilution, the backing of unbacked assets supports a fee that is accrued to the liquidity index. This fee should cover the interest earned by the minted tokens (note that the accrual to the liquidity index also redirects some interest to the minted tokens), so to offset the interest earned on *amount* it should satisfy:

$$fee \geq \begin{cases} interest, & \text{if } amount \geq supply \\ interest * \frac{supply}{supply - amount}, & \text{otherwise} \end{cases} \tag{7}$$

While the interest rate computation is part of the core protocol, the calculation of the protocol fee needed to cover unbacked amount and fee comes from outside of the core protocol. Since deferred supplies (through `backUnbacked`) and fee calculation are not enforced at the protocol level, governors should be careful in approving bridges (ports) to access the Portal feature. To prevent potential risks of excessive minting of unbacked aTokens, an `unbackedMintCap` can be specified per asset.

4.6 Price Oracle Sentinel

As noted in subsection 3.2 Aave V3 introduces a Price Oracle Sentinel to mitigate some of the UX issues that may arise in a layer 2 network.

Briefly about L2 Most layer 2 networks today (optimistic- and validity rollups) use a centralized block producer (sequencer) along with distributed validation (fraud or validity proofs) to increase throughput. Primarily these architectures support two queues for pending transactions, one being on-chain (requires L1 transaction) and another being off-chain operated by the sequencer. While the sequencer can build the next L2-"block" using transactions from both queues (with some ordering constraints for L1 not covered here), the inclusion of L1 pending transactions can often be postponed until some deadline, after which the user can force an action, be it inclusion or exodus mode for zk-sync. When the sequencer encounters downtime, this leads to the "network" not progressing the state further – no new blocks are produced. While it is still possible to send transactions to the pending transaction queues (on- or off-chain) nothing will happen immediately, the off-chain transactions may even be rejected or dropped depending on sequencer architecture and nature of downtime. Note, that even when transactions are added to the transaction queue on L1, the canonical state of the L2 is unknown until the transaction is included in a commitment to the canonical transaction history (either by sequencer or force inclusion).

For the Aave Protocol, and other systems using oracle price-feeds, this means that the feeds are *not* updated while the sequencer is down (as they use transactions after all). Essentially having a case where the entire price-development that occurred throughout the downtime is applied when the sequencer comes up. This uncertainty, and possibility for "slow flash crashes", together with the fact that queuing L2 transactions directly at L1 is out of reach for most normal users lead Aave V3 to introduce a grace-period on liquidations in these exact cases. As long as the position is not heavily undercollateralized ($0.95 < \text{HF} < 1$), it will have grace period starting at the time the sequencer comes up until it can be liquidated. If the position goes below 0.95 it can be liquidated entirely as on L1. Note, that this grace period is only activated if the sequencer has been down. During the grace period users will also not be allowed to borrow.

4.7 System roles, responsibilities and threat model

The Aave Protocol implements an access control list to segregate powers and/or benefits that can be allocated to different entities on the protocol. These roles and holders are managed in the `ACLManager` contract. The `ACLManager` keeps track of the individual roles and its holders and allows a role admin to manage roles. Role admin is itself a role that is managed by the `DEFAULT_ADMIN_ROLE`.

The contract that manages the various components of the protocol, including the `ACLManager` and the `Pool`, is the `PoolAddressesProvider`. The addresses provider keeps track of the various protocol modules and has the ability to update pointers (e.g., update `ACLManager` contract) or update the implementation of proxy contracts (e.g., update the `Pool` implementation). The `PoolAddressesProvider` is owned by the Aave Governance[5] and specifies the initial holder of the `DEFAULT_ADMIN_ROLE`. In networks other than Ethereum, either the Crosschain Governance Bridges (https://github.com/aave/governance-crosschain-bridges) or community multisigs are used to manage the `PoolAddressesProvider`.

[5] https://docs.aave.com/governance/

Role powers and responsibilities Below we outline the powers/responsibilities of the roles. The `FLASH_BORROWER` and `BRIDGE` has few direct responsibilities and can primarily access specific features of the protocol, while admin roles have the power and responsibility to handle risk or configuration parameters.

FLASH_BORROWER Holders of this role will have the premium on flash loans waived (does not include the simple flash loan).

BRIDGE Holders can perform `mintUnbacked()` and `backUnbacked()`.

ASSET_LISTING_ADMIN Holders of this role can:

Update asset oracle sources and fallback oracle.

Add new assets to the Aave market.

RISK_ADMIN Holders of this role can:

Update the grace period of Oracle Sentinels.

Update reserve parameters as reserve factor, caps, E-Mode category, borrowing on, stable borrowing allowed, freeze/unfreeze, LTV, liquidation threshold, liquidation bonus (cannot pause/unpause or activate/deactivate a reserve).

Create new and update existing E-Mode categories (not category 0).

Update unbacked mint cap and liquidation protocol fee.

ACL_ADMIN Holders of this role manage the role admins in the ACLManager.

EMERGENCY_ADMIN Holders of this role can pause and unpause the pool or an individual reserve.

POOL_ADMIN Holders of this role can update token implementations, drop, (un)pause and (de)activate reserves, update premiums along with everything the `ASSET_LISTING_ADMIN` and `RISK_ADMIN` can do.

Threat Model

Malicious Actors The following outlines the potential harm that can be caused by a malicious actor if that actor holds one of these roles.

ORACLES A malicious oracle may provide an invalid price, allowing it to borrow more than it should or perform liquidations based on the invalid price.

SEQUENCER As noted in subsection 4.6 a sequencer has significant power over ordering within the rollup, putting it in a position to extract significant value from the users of Aave when there is a significant crash in asset prices. This can be done by rejecting supply and repay actions, and liquidate users whose HF drops below 1 - effectively removing the users ability to protect themselves from liquidation in crashes.

FLASH_BORROWER If the address is a proxy, it could allow anyone to call flashloans through it to waive fees for all. This would lead to no premium to liquidity providers from flashloans.

BRIDGE If a contract/address with this role becomes malicious (or is flawed) it may mint up to the unbacked cap and never back it, effectively allowing it to steal $cap_{unbacked} - unbacked$ from the liquidity providers.

ASSET_LISTING_ADMIN An attacker can upgrade the oracle sources putting the protocol in the same state as the malicious oracle. Alternatively, the attacker may list an asset with a malicious aToken (or debttoken) implementation, allowing them to extract any funds deposited in this asset.

RISK_ADMIN The attacker can drop the Liquidation Threshold to 0 and liquidate users. This can be done atomically in the same transaction or bundle.

EMERGENCY_ADMIN The attacker can pause the pool, or unpause an insecure pool. Timed with a market crash, the attacker can turn the pool off, and then atomically perform the sequence (turn on - liquidate - turn off), allowing him to be the sole liquidator.

POOL_ADMIN The attacker may perform the attacks of the `RISK_ADMIN` or `ASSET_LISTING_ADMIN` or replace existing token implementations for new ones that include a `rug()` function only callable by the attacker.

ACL_ADMIN he attacker may give himself other roles and can then perform attacks of other admins.

ADDRESSES_PROVIDER If the owner of the addresses provider is malicious governance has fallen and it is game over.

The mitigation of the above stated "attacks" by potential malicious attackers are to be handled by Aave governance. When governance gives roles to specific actors, it should use middleware contract to limit the actions that the actor can perform using the role, e.g., for `ASSET_LISTING_ADMIN` reject updates of existing oracle feeds, and allow only additions.