Vow - Detailed Documentation

The Maker Protocol's Balance Sheet * Contract Name: * vow.sol * Type/Category: * DSS —> System Stabilizer Module * Associated MCD System Diagram * Contract Source * Etherscan *

1. Introduction (Summary)

TheVow contract represents the Maker Protocol's balance sheet. In particular, theVow acts as the recipient of both the system surplus and system debt. Its main functions are to cover deficits via debt (Flop) auctions and discharge surpluses via surplus (Flap) auctions.

?

Pictured:

- Inter-contract calls necessary for the module function
-

Not pictured:

- cage
- calls fromVow
- toFlap
- andFlop
- Auction and user auction interactions
- Governance calls / interactions
-

1. Contract Details

Vow (Glossary)

- sin
- : the system debt queue.
- Sin
- : the total amount of debt in the queue.
- Ash
- : the total amount of on-auction debt.
- wait
- : length of the debt queue
- sump
- : debt auction bid size, i.e. the fixed debt quantity to be covered by any one debt auction
- dump
- : debt auction lot size, i.e. the starting amount of MKR offered to cover thelot
- /sump
- bump
- : surplus auction lot size, i.e. the fixed surplus quantity to be sold by any one surplus auction
- hump
- : surplus buffer, must be exceeded before surplus auctions are possible
-

Other terms included in the above diagram:

- move
- : transfers stablecoin between users.
- kick
- : starts an auction.
-

Liquidations Manager

- Fess
-
    - Pushes bad debt to the auctions queue (add debt to the queue).
- Flog
-
    - Release queued debt for auction (realize debt from the queue).
- Heal
- -vow

- callsheal
- on thevat
- contract to cancel out surplus and debt. (Optimize debt buffer (vat.heal
-)).
- Kiss
-
    - Cancels out surplus and on-auction debt. Release on-auction debt and Heal (vat.heal
-).
- Flap
-
    - Trigger a surplus auction (flapper.kick
-)
- Flop
-
    - Trigger a deficit auction (flopper.kick
-)
-

Authorization

Thevow contract callskick onflop andflap to start an auction (debt and surplus auctions, respectively).

- Flopper
- (Debt auctions) - If the deficit is not covered in the forward auction portion of theflip
- auction, then debt auctions are used for getting rid of the Vow'sdebt
- by auctioning off MKR for a fixed amount of Dai. Once the auction ends, theFlop
- per will then send the received Dai to theVow
- in order to cancel its debt. Lastly, theFlop
- per will mint the MKR for the winning bidder.
- Flapper
- (Surplus auctions) - These are used for getting rid of theVow
- 'ssurplus
- by auctioning off a fixed amount of internal Dai for MKR. Once the auction ends, theFlap
- per burns the winning MKR bid and sends internal Dai to the winning bidder.
-

SystemData

- System config
- Vow.wait
-
    - Flop delayVow.sump
-
    - Flop fixed bid size
- Vow.dump
-
    - Flop starting lot sizeVow.bump
-
    - Flap fixed lot sizeVow.hump
-
    - Surplus buffer
-

Debt (SIN) Queue

When a Vault is liquidated (bite -documentation), the seized debt is put in a queue for an auction in aVow (labeled assin[timestamp] - the system debt unit). This occurs at the block timestamp of thebite action. It can be released for auction viaflog (flog releases queued debt for the auction) once the allottedVow.wait (the flop delay) time has expired.

TheSin is stored when it's in the debt queue, but the debt available to auction isn't explicitly stored anywhere. This is because the debt that is eligible for auction is derived by comparing theSin (i.e. debt on the holding queue) with the dai balance of theVow as recorded inVat.dai[Vow] . For instance, ifVat.sin[Vow] is greater than the sum ofVow.Sin and theAsh (debt currently on auction), then the difference may be eligible for aFlop auction.

Notes:

- In the case of when acat.bite
- /vow.fess
- is executed, the debttab

- is added tosin[now]
- andSin
- , which blocks thattab
- amount to be sent to theflop
- auction and all of the DAI is recovered with aflip
- auction. In theory, unblocking thetab
- amount in theSin
- shouldn't be necessary, but in practice it actually is. If this debt is not unblocked, then when we have a real need to send aflop
- auction, we might have a bigSin
- that blocks it. To summarize, this means that each registry ofsin[era]
- that has an amount > 0 should beflog
- 'ed before kicking aflop
- auction (this is because in order to kick the whole thing, you need every register to be 0, otherwise, it would be blocking debt).
-
    - Eachsin[era]
-
    - isn’t required to be a singlebite
-
    - , it will group all thebite
-
    - ’s that are in the same Ethereum block together.
-
    - Theauction-keeper
-
    - willflog
-
    - everyera
-
    - with positiveSin
-
    - if thewoe
-
    - +Sin
-
    - =sump
-
    - , wherewoe
-
    - =vat.sin[vow]
-
    - -vow.Sin
-
    - -vow.Ash
-
    - .
-
    - Where the components withinvat.sin(vow)
-
    - -vow.Sin
-
    - -vow.Ash
-
    - are defined as:
-
        - vat.sin(vow)
-
        -
            - total bad debt
-
        - vow.Sin
-
        -

        - debt blocked
-
    - vow.Ash
-
    -
        - debt in auctions

- *
- Vow.sin
- records individual portions of debt (marked with a timestamp). These are not directly auctioned off, but cleared whenflog
- is called.
- If theSin
- is not covered by holding aflip
- auction within the designated wait time (tau
-), theSin
- “matures” and gets marked as bad debt to theVow
- . This bad debt can be covered through a debt auction (flop
-) when it exceeds a minimum value (thelot
- size). In short, the time between the debt being added to thesin[]
- queue and becoming "mature" (when itflog
- s off the queue and is eligible forFlop
- auction) is the amount of time thatFlip
- auction has to clear that debt. This is due to the fact that when aFlip
- auction receives DAI, it decreases theVow
- 's DAI balance in theVat
- .
- Note:
- In this case, there is a risk that a circumstance can occur where theVow.wait
- is different than theFlip.tau
- . The main risk being related towait
- <tau
- , which would result in debt auctions running before the associated seized-collateral auctions could complete.
-

Overall Sin can affect the system in the following way:

1. There can be separateVow
2. s each with their ownsin
3. s
4. In the case of an upgrade, if we remove aVow
5. that hassin
6. , this can create untracked bad debt in the system.
7.

Accounting

Vow.Sin - This calculates the total queued debt in the system.Vow.Ash - This calculates the total on-auction debt.

1. Key Mechanisms & Concepts

It is important to note that the Maker Protocol will deviate from its equilibrium. This occurs when it receives system debt and system surplus through the collateral auctions and Vault stability fee accumulation. TheVow contract contains the logic to trigger both the debt (flop) and surplus (flap) auctions, which work to correct the system’s monetary imbalances.

Summary

- System Debt:
- In the case where Vaults are bitten (liquidated), their debt is taken on by theVow
- contract as aSin
- (the system debt unit). TheSin
- amount is then placed in theSin
- queue.Note:
- When theSin
- is not covered by aflip

- auction (within the dedicatedwait
- time, theSin
- is considered to have bad debt to theVow
- . This bad debt is then covered through a debt auction (flop
-) when it exceeds a minimum value (thelot
- size).
- System Surplus:
- Occurs from stability fee accumulation, resulting in additional internal Dai in theVow
- . This surplus is then discharged through a surplus auction (flap
-).
-

1. Gotchas (Potential source of user error)

2. When theVow

3. is upgraded, there are multiple references to it that must be updated at the same time (End
4. ,Jug
5. ,Pot
6.).
7. TheVow
8. is the only user with a non-zeroSin
9. balance (not avat
10. invariant as there can be multipleVow
11. s).
12. Ilk storage is split across theVat
13. ,Jug
14. ,Pot
15. andVow
16. modules. Thecat
17. also stores the liquidation penalty and maximum auction size.
18. A portion of the Stability Fee is allocated for the Dai Savings Rate (DSR) by increasing the amount ofSin
19. in theVow
20. at everyPot.drip( )
21. call.
22. Setting an incorrect value forvow
23. can cause the surplus to be lost or stolen.
24.

1. Failure Modes (Bounds on Operating Conditions & External Risk Factors)

Vault Liquidation

- A failure mode could arise when no actors callkiss
- ,flog
- orheal
- to reconcile/queue the debt.
-

Auctions

- A failure mode could arise if a user does not callflap
- orflop
- to kick off auctions.
- Vow.wait
- , when set too high (wait
- is too long), theflop
- auctions can no longer occur. This provides a risk of undercollateralization.
- Vow.wait
- , when set too low, can cause too manyflop
- auctions, while preventingflap
- auctions from occurring.
- Vow.bump
- , when set too high, can result in noflap
- auctions being possible. Thus, if noflap
- auction takes place, there will be no MKR bidding as part of that process and, accordingly, no automated MKR burn as a result of a successful auction.
- Vow.bump
- , when set too low, results inflap

- auctions not being profitable for participants (lot
- size is worth less than gas cost). Thus, no MKR will be bid during aflap
- auction and, as a result, there will be no automated MKR burn.
- Vow.sump
- , when set too high, noflop
- auctions are possible. This results in the system not being able to recover from an undercollateralized state.
- Vow.sump
- , when set too low,flop
- auctions are not profitable for participants (where thelot
- size is worth less than gas cost). This results in MKR inflation due to automated MKR minting.
- Vow.dump
- , when set too high,flop
- auctions risk not being able to close or mint a large amount of MKR, creating a risk of MKR dilution and the possibility of a governance attack.
- Vow.dump
- , when set too low,flop
- auctions have to bekick
- ed many times before they will be interesting to keepers.
- Vow.hump
- , when set too high, theflap
- auctions would never occur. If aflap
- auction does not occur, there is no sale of surplus, and thus, no burning of bid MKR.
- Vow.hump
- , if set too low, can cause surplus to be auctioned off viaflap
- auctions before it is used to cancelsin
- from liquidations, necessitatingflop
- auctions and making the system run inefficiently.
-

Previous Flopper - Detailed Documentation Next Oracle Module Last updated4 years ago On this page *

Export as PDF