Pause - Detailed Documentation

A delegatecall based proxy with an enforced delay * Contract Name: * pause.sol * Type/Category: * Governance Module * Associated MCD System Diagram * Contract Source *

1. Introduction (Summary)

Theds-pause is adelegatecall based proxy with an enforced delay. This allows authorized users to schedule function calls that can only be executed once a predetermined waiting period has elapsed. The configurable delay attribute sets the minimum wait time that will be used during the governance of the system.

?

1. Contract Details:

Key Functionalities (as defined in the smart contract)

Plans A plan describes a singledelegatecall operation and a unix timestampeta before which it cannot be executed.

A plan consists of:

- usr
- : address to delegatecall into
- tag
- : the expected codehash of usr
- fax
- : calldata to use
- eta
- : first possible time of execution (as seconds since unix epoch)
-

It is important to note that each plan has a unique id, defined as a keccack256(abi.encode(usr, tag, fax, eta)).

Operations

Plans can be manipulated in the following ways:

- plot
- : schedule a plan
- exec
- : execute a scheduled plan
- drop
- : cancel a scheduled plan
-

Thepause contract contains theDSPauseProxy contract in order to allow plan to be executed in an isolated storage context to protect the pause from malicious storage modification during plan execution.

1. Key Mechanisms & Concepts

Theds-pause was designed to be used as a component in the Maker Protocol’s governance system in order to give affected parties time to respond to decisions. If those affected by governance decisions have e.g. exit or veto rights, then the pause can serve as an effective check on governance power.

1. Gotchas (Potential source of user error)

Identity & Trust

In order to protect the internal storage of the pause from malicious writes during plan execution, we perform thedelegatecall operation in a separate contract with an isolated storage context (DSPauseProxy), where each pause has its own individual proxy.

This means that plans are executed with the identity of theproxy . Thus when integrating the pause into some auth scheme, you will want to trust the pause's proxy and not the pause itself.

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

A break of any of the following would be classified as a critical issue:

High level

- There is no way to bypass the delay
- The code executed by the delegatecall cannot directly modify storage on the pause
- The pause will always retain ownership of it's proxy
-

Administrative

- authority, owner, and delay can only be changed if an authorized user plots a plan to do so
-

Plot

- A plan can only be plotted if its eta is after block.timestamp + delay
- A plan can only be plotted by authorized users
-

Exec

- A plan can only be executed if it has previously been plotted
- A plan can only be executed once it's eta has passed
- A plan can only be executed if its tag matches extcodehash(usr)
- A plan can only be executed once
- A plan can be executed by anyone
-

Drop

- A plan can only be dropped by authorized users
-

Other Failure Modes

DSPause.delay - when the pause delay is set to the maximum, governance can no longer modify the system.

DSPause.delay - when the pause delay is set to the minimum, it is easier to pass malicious governance actions.

Previous Spell - Detailed Documentation Next Chief - Detailed Documentation Last updated4 years ago On this page *1. Introduction (Summary) * 2. Contract Details: * 3. Key Mechanisms & Concepts * 4. Gotchas (Potential source of user error) * 5. Failure Modes (Bounds on Operating Conditions & External Risk Factors)

Export as PDF