# Main concept and components of the proposed work:

The proposal focuses on the research and provision of a reference implementation framework for smart contract recovery –with a focus on smart contracts used in non-financial applications. In other words, we are looking to produce a smart contract recovery module, made up of different recovery mechanisms, similar to Open Zeppelin’s work for access: <https://github.com/OpenZeppelin/openzeppelin-contracts/tree/master/contracts/access>

The reference implementation would be made up of multiple recovery mechanisms (based on our research into existing, robust implementations), including a novel concept and architecture we call: **Recovery Pulse.**

The proposal aims to provide a new, UX-friendly way of doing smart contract and/or account and/or data recovery on the Ethereum blockchain and by extension, any EVM chain. This grant would fund our research efforts and implementation

The **Recovery Pulse** mechanism is one of the approaches we have conceptualised and will be looking to offer in this reference implementation as a generally applicable mechanism for non-financial applications. This mechanism is based on a smart contract architecture that would enable transfer of ownership over a contract based on either of these two possible events:

* An internal “Pulse” transaction being emitted from the contract owner every ***t*** amount of time to prove that the contract owner is still actively in control of the contract and does not require help recovering it.
* An external “Pulse” transaction being triggered by 1 of ***n*** addresses (nominated by the smart contract owner). This externally triggered transaction would be “pending” and executed only once the ***t*** threshold has been reached and no “Pulse Proof” has been triggered by the owner of the contract.

The scope of this proposal covers the research of existing social recovery implementations in non-financial applications, development of a proof of concept implementation of the Recovery Pulse mechanism in smart contract, as well as the testing and documentation of the implementation.

This smart contract/account recovery mechanism does not require explicitly setting up guardians and distributing keys to various individuals. The trigger and recovery events could be independent from an explicit set-up process required from the smart contract deployer address.

# Initial review and research

## Summary of general concepts: Multisigs-based accounts, MPC-based accounts, social recovery:

Three general approaches to recovery have been used by web3 users. The first involves using **Mulitisgs** to secure multi-address access to one’s assets and require a signature threshold to be reached. The popularity of the Safe protocol is in part due to the effectiveness of this approach and simple set-up. It can be summarised as follows:

A diagram of a wallet contract

Description automatically generated

Source: Vitalik Buterin (2021)[[1]](#footnote-0)

The second approach involves **accounts based on** **Multi Party** Computation (e.g., ZenGo wallets). Under this paradigm, we create and distribute *shares* of a private key such that no one single person or machine controls the private key entirely — this process is called Distributed Key Generation (DKG):

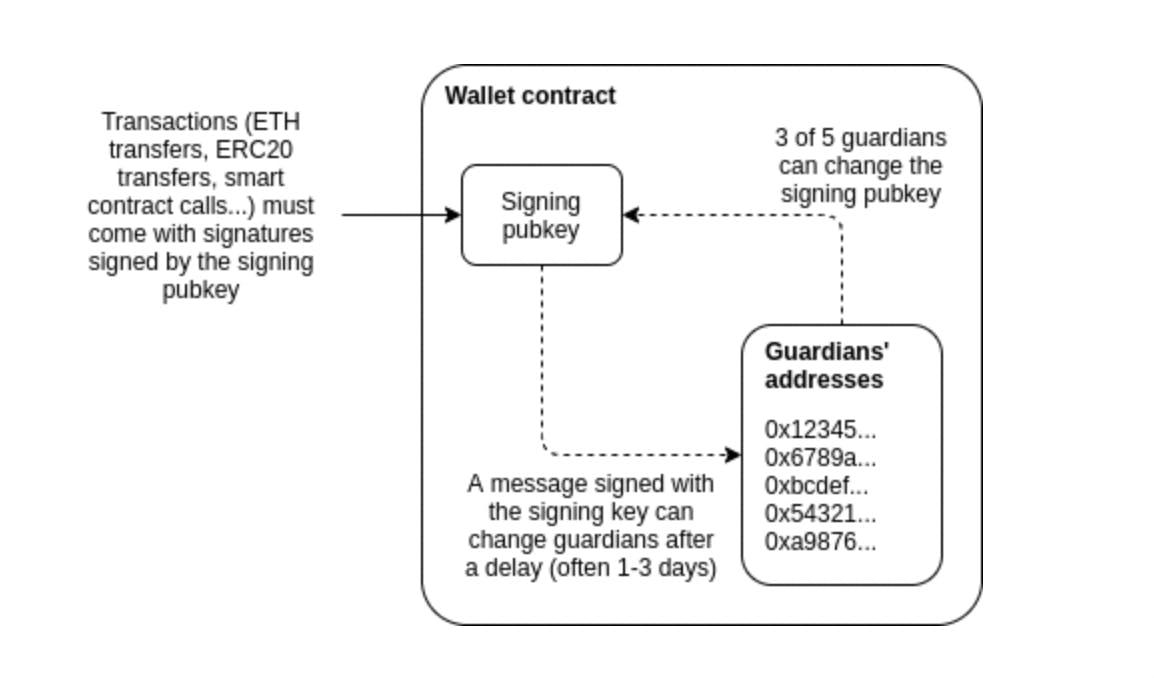
* Individual secrets are randomized by each of the several (always more than 3) endpoints – either servers or mobile devices. Those secrets are never shared with each other.
* The individual endpoints engage in a decentralized wallet creation protocol in which they compute the public key (wallet address) that corresponds to the set of individual private shares.
* When a signature on a blockchain transaction is requested, a quorum (at least 3) of endpoints engage in a distributed signature process where each one of the endpoints individually validates the transaction and policy and signs the transaction.[[2]](#footnote-1)

A diagram of a wallet

Description automatically generated

Compared to multisigs, MPC-based accounts have five practical characteristics that can affect the account recovery experience: Adjustable Signing Schemes; can modify quorum without changing addresses; key refresh; lower onchain recovery costs; blockchain agnostic.; (potential drawback) offchain accountability, i.e., signing authorization policies and approval quorums are managed offchain and subject to centralized failures.

## The final approach is **social recovery** for account and smart contract recovery (rising popularity with wallets like **Argent** leading the way). This approach has spawned a diverse range of sub-designs that focus on different verification policies and requirements that we explore in the next section:



1. There is a single "signing key" that can be used to approve transactions.
2. There is a set of at least 3 (or a much higher number) of "guardians", of which a majority can cooperate to change the signing key of the account.

The signing key has the ability to add or remove guardians, though only after a delay (often 1-3 days). Guardians do not have to be publicly known: in fact, they do not need to know *each other's* identities.Each guardian can be asked to deterministically generate a new single-purpose address that they would use just for that particular recovery; they would not need to actually send any transactions with that address unless a recovery is actually required.   
Source: Vitalik Buterin (2021)[[3]](#footnote-2)

# Use cases for the decentralised recovery of smart contracts and accounts (i.e., secret keys):

### Higher standard digital asset protection for organisations

### Public DAO recovery mechanism enforced by a community or core contributors

### Inheritance

### Whistleblowing (specifically non-financial)

### Emergency Communication (specifically non-financial)

### Information archiving and preservation (specifically non-financial)

### Time Capsules (specifically non-financial)

# Selection criteria for our research:

* POC vs live in production
* Level of usage
* Functional differences with other approaches
* Architectural and operational characteristics
* Lindy / robustness
* Set-up difficulty
* Audit level
* Attack vectors / known exploits
* Threshold of risk (spectrum of risk within financial vs non-financial applications)

# Notable projects and architectures:

### **Argent** social recovery

* Guardians-based recovery
* Time-delay for guardian changes
* In production and used by thousands of users (high popularity)
* Multiple audits
* At least 3 years old
* No known exploit
* Easy set-up but costly
* Risk threshold adapted to financial assets and recovery

A computer screen shot of a program

Description automatically generated

Source: <https://github.com/argentlabs/argent-contracts-l2/blob/master/contracts/ArgentWallet.sol>   
  
Argent also offers an offchain approach to recovery on generation of a random “key-encryption-key” (KEK) that encrypts the user’s private keys. The encrypted private keys are uploaded to iCloud, and the KEK — which is required for decryption — is sent to the Argent servers.  
  
Source: <https://www.argent.xyz/blog/off-chain-recovery/>

### Loopring social recovery

* Guardians-based recovery
* Modifiable time-delay for guardian changes
* In production (low popularity)
* Multiple audits
* At least 2 years old
* No known exploit
* Easy set-up and low cost on native L2
* Risk threshold adapted to financial assets and recovery

Uses InheritanceLib

A screen shot of a computer screen

Description automatically generated

Source: <https://github.com/Loopring/protocols/tree/release_loopring_3.6.2/packages/hebao_v2>

1. **Safe Modules:**We highlight four Safe Modules for enabling recovery mechanisms on the Safe multisigs, ranging in their policies, need for guardians and notification of parties during recovery events (an important issue as it can be a point of failure/attack in social recovery approaches). **a. Proof of concept –** Delay-based social or public recovery by Philogy.   
   Summarised approach as:
   * Safes can assign a list of recovery agents and their respective delays
   * Recovery agents can only recover the safe after it has been inactive for the duration of their delay
   * Each recovery agent can have a custom delay (allows expressing recovery priority preferences)
   * Safe inactivity is tracked via the last authorized transaction
   * Upon recovery the recovery agent will replace the owners of the safe
     + **Mechanics (Storage)**
   * For each safe the recovery contract will keep track of the following data:
   * recoveryAgentRoot: Merkle tree of approved recovery addresses and their recovery delay
   * ping: timestamp when safe last executed an authorized transaction
   * recoveryAgent: Currently active recovery agent, address(0) if none
     + **Logic - Pinging**
   * As a safe’s guard the recovery contract will update the safe’s ping when it’s triggered with either the checkAfterExecution or checkTransaction hooks.
     + **Logic - Recovery**
   * To initiate recovery the calling account will pass the merkle proof along with a new set of owners to replace the old.

Source: <https://forum.safe.global/t/social-recovery-module/2117/4>

**b. Proof of concept – Inheritence** approach by Kevin Foes.   
Highlight on the fact that guardians are not required for recovery here. This canfacilitate public recovery mechanisms.  
Lower risk threshold – ideal for recovery of accounts preserving public data.

A screen shot of a computer program

Description automatically generated

Source: <https://github.com/keviinfoes/Safe-recovery-modules/blob/main/contracts/InheritanceModule.sol>

1. **Proof of concept – Secret recovery** by Kevin Foes.   
   Highlight on the requirement for Guardians and generation of custom secrets.

A screen shot of a computer program

Description automatically generated

Source: <https://github.com/keviinfoes/Safe-recovery-modules/blob/main/contracts/SecretRecoveryModule.sol>

1. **Proof of concept** – Recovery with in-built notification service by Benjamin x:

A screen shot of a computer

Description automatically generated

Source:

<https://goerli.etherscan.io/address/0x5616cc556C32AafDB46E5F7018c52b41bB20Cd90#code>   
<https://dev.to/bxmmm1/safe-recovery-module-1mf>

### **Proof of concept –** Anti-Social Recovery by EY:

* Self-recovery based on verifiable proof of humanity.
* Require no guardian.
* Basic audit via EthGlobal competition.
* Verify unique proof-of-personhood with Worldcoin World ID
* Deploy a smart contract where the proof-of-personhood serves as an input to prove ownership.
* High risk threshold, where owner is the only legitimate party.

**Example functions and methods**

A screenshot of a document

Description automatically generated

Source: <https://github.com/crystaltai/anti-social-recovery-club-2>

### StarkNet – Switch recovery service

* Based on storage proofs (Oiler Network)
* Recovery of EOA instead of a contract
* Chain-native
* Time-based rules
* In production (low popularity)
* Unknown audits
* At least 1 year old
* No known exploit
* Medium risk threshold

**Description: Switch** is a fully trustless wallet recovery service for Ethereum Externally Owned Accounts (EOA - your normal Ethereum address, not smart contract wallet), powered by storage proofs on StarkNet. Unlike social recovery and other off-chain methods for wallet retrieval, Switch runs in an entirely trustless, non-custodial way. This allows users to build more fault tolerant wallet setups, while retaining the strong security guarantees of Ethereum.

To recover user funds, Switch uses storage proofs (by Oiler Network) to recursively derive the past state of an EOA and determine whether an account has been lost. This works (broadly) as follows:

* The owner of an EOA delegates control of their funds to a RecoveryContract on L1.
* Access details for EOA are lost.
* After a set period of inactivity (say 1 year), the user can call a StorageProver contract on StarkNet to verify that the nonce of the lost EOA has not changed over the period. If the test passes, the EOA is treated as lost and the L1 RecoveryContract is notified.
* To recover their assets, the user pings the L1 RecoveryContract to withdraw all delegated assets.
* Switch is currently deployed for Ethereum L1 but can be ported to any other L1 blockchain which uses Patricia Merkle Trees.

Source: <https://github.com/Starknet-Recovery-Service>

### ZenGo – Secure Recovery

* Three-way recovery – file, biometric, social account
* Modifiable time-delay for guardian changes
* In production (high popularity - 1m users)
* Multiple audits
* At least 3 years old
* Medium difficulty set-up and recovery
* No known exploit
* Partially proprietary
* **High risk threshold – institutional assets**

Three recovery approaches:

* Zengo app creates a unique recovery file (recovery kit) and stores it in your default personal cloud storage system, depending on your operating system (Google Drive for Android and iCloud Drive for iPhone).
* Your 3D FaceLock is an anonymous, mathematical representation of your face used to authenticate you. It is performed **locally** and **privately** on your phone.
* Your email account Something you store: The Zengo recovery kit, stored securely in at least one of your cloud backup locations.

**+ Proof of concept – Legacy transfer by ZenGo**:  
A self custodial, inheritence-style feature. Supports user initiated and chain agnostic transfers after prolonged period of account inactivity.  
 Source: <https://github.com/ZenGo-X/public-docs/blob/main/Zengo%20Legacy%20Transfer%20White%20Paper.pdf>

1. **Farcaster**

In production

* Guardians-based recovery
* Instant process / delay-based
* In production (medium-level popularity)
* Multiple audits
* At least 0.2 years old
* No known exploit
* Easy set-up and low cost on Optimism L2
* Medium risk threshold adapted to non-financial

**Description**: Fids have a time-delayed recovery system which protects against the loss of the custody address. The recovery system works like this: The custody address can nominate a recovery address at any time. The recovery address can request a transfer, which will move the fid from the custody address to a new address after a delay. For 3 days after the request, the custody address can cancel the request made by the recovery address. After 3 days, the recovery address can complete the transfer, will moves the fid from the custody address to the new address. The delayed recovery allows users to choose semi-trusted partners for recovery such as a third-party service or a distant friend. If the service initiated a request without the user's approval, they could always cancel it.

Source: <https://github.com/farcasterxyz/protocol/discussions/100>

1. **POC** Token-specific recovery by Martins et al.:

* Token-specific recovery
* Architecture that can be replicated in other contracts (smart accounts potentially).
* Requires blockchain-based dispute resolution mechanism i.e., can be an architecture for public recovery and deliberation rather than based on closed social circles.
* Uses IPFS as offchain storage service

Source: <https://www.dpss.inesc-id.pt/~mpc/pubs/Recoverable_Token__Filipe_Martins_final.pdf>

1. **POC** privacy preserving recovery by ZKW:

* Built on top of Semaphore protocol
* Privacy preserving – hides the identity of the guardians and main   
  Source <https://github.com/zkSocialRecoveryWallet/zkSocialRecoveryWallet>

1. **Recovery interface by Zhang et al.:**

Interfaces for social recovery account supporting various guardian types and customizable recovery policies. This is an ERC that proposes a standard interface for social recovery of smart contract accounts. It separates identity and policy verification from the recovery process, allowing more ways to authenticate (known as Guardians) than just on-chain accounts. It also lets users customize recovery policies without changing the account’s smart contract.

1. Appoint friends or family members, who do not have blockchain accounts, as Guardians for social recovery.
2. Use NFTs/SBTs as Guardians for their accounts.
3. Personalize and implement adaptable recovery policies.
4. Support novel types of Guardians and recovery policies without needing to upgrade their account contracts.
5. Enable multiple recovery mechanism support, thereby eliminating single points of failure.

The account may implement the IRecoveryAccount interface to support social recovery functionality, enabling users to customize configurations of different types of Guardians and recovery policies. In the contract design based on Module, the implementation of RecoveryModule is very similar to RecoveryAccount, except that different accounts need to be distinguished and isolated.

A screen shot of a computer code

Description automatically generated

**Source:** [**https://eips.ethereum.org/EIPS/eip-7093**](https://eips.ethereum.org/EIPS/eip-7093)

# Related tech, i.e., core components and primitives that are used in at least one of the specified approaches:

* Reference functions / events / errors for access controls and ownership transfers: <https://docs.openzeppelin.com/contracts/5.x/api/access>
* Shamir secret sharing  
  <https://slips.readthedocs.io/en/latest/slip-0039/>
* Safe Protocol
* Threshold Signature Scheme (TSS)
* Ethereum Gas Network Station ( Necessary for improving the UX of recovery spread across non-technical users)  
  <https://docs.opengsn.org/>
* Web3Auth: <https://web3auth.io/docs/what-is-web3auth#what-does-web3auth-do>
* [EIP-4337: Account Abstraction via Entry Point Contract specification](https://eips.ethereum.org/EIPS/eip-4337)
* [EIP-6900: Modular Smart Contract Accounts and Plugins](https://eips.ethereum.org/EIPS/eip-6900)

# Recovery pulse:

1. **Web2 solutions:**
   * + - **Google** created a DMS for accounts called Inactive Account Manager. It allows the account holder to nominate someone else to access their services if not used for an extended period (the default is three months).
       - **Facebook** has a “Legacy Contact” feature that allows users to designate someone to manage their account after they die.
       - **Microsoft** has a “Next of Kin” feature for email accounts.
       - **Instagram** has a “Memorialization” feature
2. **Sarcophagus** – Dead Man Switch DAO

* Dead Man’s Switch model with decentralised node operators and a DAO.
* In production (low popularity - <https://dune.com/sliceanalytics/sarcophagus-network> )
* Token and network based
* Off chain storage used
* At least one audit
* A switch that triggers at t=365 where t the number of days since activity has been detected from the key pair of the address
* Immutability and data permanence.
* Varying risk thresholds – multiple use cases

**Description**: Sarcophagus is designed to allow recovery at custom intervals, when the payload is available for decryption based on a lack of response from the creator. A more complete backup system than traditional recovery mechanisms. Immutable data delivery via automated contracts ensures that data makes it to the recipient when the time comes without any third party assistance or interference. All data stored is encrypted so only the creator knows what is in a Sarcophagus until it is released. Using Arweave, data is permanent which guarantees it will be available when the DMS triggers.

Sources: <https://github.com/sarcophagus-org/sarcophagus-contracts> <https://github.com/sarcophagus-org/quickstart-archaeologist>

1. **Proof of concept –** Timelock by Peter Todd

* Made for Bitcoin addresses
* Uses parallel serial hash chains
* No known audit
* High risk threshold

**Description**: Create a secret key that can be decrypted in a known amount of time using parallel-serial hash chains. The creator can compute the timelock in parallel, taking advantage of the large amount of cheap parallelism available today, while others are forced to compute it serially, constrained by the lack of scalar performance growth.

The chains are constructed such that Bitcoin addresses can be derived from them and bounties placed, incentivizing third-parties to crack the timelocks. This gives us a valuable chance to incentivize others to push the envelope of scalar performance - important knowledge if we are going to have any hope of knowing how soon our timelocked secrets will actually be revealed! The Bitcoin secret keys and addresses are constructed from the chains as follows:

Source: <https://github.com/petertodd/timelock>

1. TEIA – Dead man’s switch

* At least 0.5 years old
* Tezos blockchain
* Low popularity
* Contract ready

Description: A dead man's switch contract. A multisig contract will take control of the funds in case the contract admin is dead or lost access to their keys. The admin needs to ping the contract regularly to indicate that they are still alive.

A screenshot of a computer program

Description automatically generated

Source: <https://github.com/teia-community/teia-smart-contracts/blob/main/python/contracts/deadMansSwitch.py>

1. **Killcord – dead man’s switch**

* Contract ready
* No audit
* At least 3 years ago
* Uses IPFS for offchain storage
  + Client generates necessary files (including keys and payloads).
  + Encrypted payload is placed on IPFS.
  + Keys are placed on a trusted published (potentially single point of failure).
  + A smart contract running on the EVM continuously checks for pings from clients. If client doesn't check in over some pre-defined policy, then trusted published will be aware and publish keys to the smart contract, visible to everyone.

Description: Killcord is a tool used to build resilient deadman's switches for releasing encrypted payloads. In its default configuration, killcord leverages ethereum and ipfs for censorship resistance. The killcord project owner hides a secret key from the world by **checking in** to the killcord smart contract on ethereum. If the owner stops checking in after a period of time, the killcord is triggered and the secret key that decrypts an encrypted payload is published.

The easiest way to get started with killcord is by **watching** an existing project and decrypting its payload.

A screen shot of a computer code

Description automatically generated

Source: <https://github.com/nomasters/killcord/blob/master/contract/killcord.sol>

1. https://vitalik.ca/general/2021/01/11/recovery.html [↑](#footnote-ref-0)
2. https://www.fireblocks.com/blog/mpc-vs-multi-sig/ [↑](#footnote-ref-1)
3. https://vitalik.ca/general/2021/01/11/recovery.html [↑](#footnote-ref-2)