**IMPLEMENTATION**

### IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it’s constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

**MODULS:**

1. **Som: shared-ownership file access control model**
2. **Commune: distributed enforcement of shared ownership**
3. **Collusion resistant secret sharing (crss)**
4. **Comrade: blockchain-based shared ownership**

**MODULE DESCRIPTION**

1. **Som: Shared-Ownership File Access Control Model:-**

In this module, we define the concept of shared ownership, and formally instantiate it in a file access control model dubbed SOM. Our main motivation for constructing this model is three-fold:(i) to precisely define the ideal set of features that we believe a model, which enforces shared ownership, should provide; (ii) to formulate the problem of distributed enforcement more precisely by focusing on SOM’s formal description; (iii) to provide a point of reference to scrutinize SOM’s enforcement solutions, including our own.

1. **Commune: Distributed Enforcement Of Shared Ownership:-**

This section presents Commune, our solution for distributed enforcement of the SOM access control policy in an agnostic cloud. As SOM does not specify concrete file access operations, we instantiate Commune with write and read actions. Before introducing our solution, we outline our cloud and attacker model.

1. **Collusion Resistant Secret Sharing (Crss)**

We now introduce our second building block, called Collusion Resistant Secret Sharing (CRSS). Similar to threshold secret-sharing schemes, CRSS allows one party to distribute a secret among a set of designated shareholders, so that any subset of shareholders of size equal to or greater than the threshold can reconstruct the secret. Furthermore, CRSS allows shareholders to issue to other users delegation to reconstruct the secret. If a user collects enough (i.e., above the threshold) delegations, he can rightfully reconstruct the secret. However, users cannot pool their delegations to reconstruct the secret, unless one of them has collected enough delegations. In Commune, CRSS is used to secret-share the key K used in SFD, in order to achieve collusion resistance. CRSS is inspired by decentralized Attribute Based Encryption [18] where shares of a secret are blinded with shares of 0, such that, if a user collects enough shares for his identity, the blinding cancels out and the secret can be reconstructed.

1. **Comrade: Blockchain-Based Shared Ownership:-**

In this section, we present an alternative solution for enforcing shared ownership in the cloud by leveraging functionality from the blockchain. Our solution, dubbed Comrade, enables a distributed blockchain-based enforcement of the SOM access control policy in a cooperative cloud. Unlike Commune, Comrade does not assume an agnostic cloud, and requires the cloud operator to co-operate and to interface with the blockchain. Since SOM does not specify concrete file access operations, we instantiate Comrade with write and read actions. Before introducing our solution, we provide some background on the blockchain and describe the system model.