I propose a Byzantine fault tolerant system, that is, a (highly) distributed system, with critical system records and ledgers stored on a block chain distribution method ,)think torrents_users one user building a file via a combination of inputs from a distribution of sources) (such as that individual nodes/users/hold/share and cross verify records, such as that all records, including addition records or any other change to any, effectively, ‘data base’ cannot be compromised in the sense that the permissions, an agreement, is between that and the public, but is itself a public record, wholly scrutinizable, and unaffected by tampering.

As i build the democrat.computer
I would like to include this

A record like this would absolutely really protect the party from any tampering, part of this type of ledger system, one key, is distributing freely and openly as much as possible thus protecting the core data- system information needing security-- whereas, depending of the content/ can be a major part of the content. [these are public records] Private records are distributed in a way, that a collection of processes ensures authority in the sense of what changes are made to current records, the addition of records, and the process over which data and modification to it and rules are governed,

And this is not to ensure any given version could replace a set of data in a catastrophe, say one dic-connectiond the nodes/chain of agents who all host complete or partial copies of any/all record(s); the ‘fecundity of the system’ the ability to be replicated is not affected by a global catastrophe. Durbal system are carefully versioned and maintained and should never be compromised. [the empty data record could be called a shell record: a group of ‘sockets’ locations for data, which (data) is not part of the record of the architecture by design] thus the combination of these 2 things, (authoritative records) data, and architecture (software), entail the necessity of “work” distribution across them , to be necessary. Deontologically purposively,