To our knowledge, this is the only work combining the idea of a managed cryptocurrency with the open consensus model used by unmanaged currencies. The work most similar to ours is Multichain. It provides a platform for creating and deploying ‘private’ blockchains within or between organizations. It is designed to provide the following features :

Bitcoin has high transaction fees due to limits on transaction throughput, but this is a technical problem not necessarily present in other cryptocurrencies.

1) ‘to ensure that the blockchain’s activity is only visible to chosen participants’

2) ‘to introduce controls over which transactions are permitted’

3) ‘to enable mining to take place securely without proof of work and its associated costs’

Instances of Multichain have an administrator or group of administrators that define the ongoing policy of the system. They have complete control in defining who can view the blockchain, who can put transactions on the blockchain, and who can maintain the blockchain (those mining new blocks). This last feature enables them to maintain the blockchain at very little cost since the computationally expensive proof-of-work consensus methods of Bitcoin can be dispensed with. This is replaced with a flexible round robin approach where the miners mostly take turns publishing the new blocks and generally do not receive any reward for doing so (since the work is trivial).

While a powerful approach for organization-run blockchains, Multichain cannot be used to satisfy our stated objectives since the administrators have complete control. There is no mechanism to implement a balance of power where the administrators can manage the currency in an ongoing fashion but where the maintainers of the blockchain can ensure that the administrators follow the stated rules of the cryptocurrency.

MANAGED CRYPTOCURRENCY ARCHITECTURE

All blockchains contain a ‘genesis block’. This is the first block on the blockchain and it has no pointer to a previous block (being the first one). All users of the blockchain must agree on this first block for a consistent view of the blockchain to exist. We propose the addition of a ‘genesis transaction’5. This is the first transaction in the blockchain and it defines an account that has the currency manager role (and is owned by the currency administrator). In our system, only accounts with roles can issue transactions and only accounts with the currency manager role can create other accounts with roles (with one important exception, discussed later). Thus, the genesis transaction is the transaction that enables all other transactions.

The initial account is the root of a hierarchical tree of nodes, where each node represents an account labeled with a set of roles6. The root node not only has the currency manager (M) role7, but it has all other available roles: central banker (C), law enforcement (L), user (U), and account manager (A).We label the roles of an account by concatenating all applicable labels. Thus, the root node has the role set ‘MCLUA’.

When a node with the M role creates a new account (more precisely, it labels some unlabeled account created by some user), it bestows on that account a, not necessarily proper, subset of its roles. Thus, the cardinality of the set labels for nodes monotonically decreases as one traverses higher in the hierarchy tree. One exception to this monotonicity rule is that nodes with the M label may also modify the role sets of nodes higher in the tree (provided they are on the path from the target node to the root), restricted again to the set of roles possessed.

Nodes with the A role may also create and delete accounts, but such created accounts may only have the U role. The currency administrator then can delegate user account management to third party organizations by giving them the A role.

The different roles provide different accounts different capabilities:

This is related to the ”asset genesis” metadata transaction idea but is more powerful as it controls all transactions on the blockchain.

We use the terms node and account interchangeably depending upon the desired perspective (node in a tree versus account owned by a user)

The M role is distinct from the currency administrator. Many accounts may have the M role but there exists a single entity which is the currency administrator.

The U role enables an account to receive and spend coins. An account for which the U role has been removed has its funds frozen.

The A role enables an node to create accounts with the U role (and only the U role). It may also remove the U label for its descendants.

•

TheCrole enables the creation of new coins (apart from the block mining rewards).

•

The L role enables an account to forcibly move funds between accounts, to remove the U label, and to restore a previously removedUlabel.However, these actions can only be performed against nodes with the same or greater distance from the root.

The currency administrator, who will own the root M labelled node, may require thatAnodesverify users’ identities prior to providing an account. In this case, the architecture enables a system where the ‘know your customer’ (KYC) laws might be satisfed. Individual transacting parties would not know each other’s identitiesbut some account authorizing entity would have a record for each account with the U role. Fulflling KYC laws is a general problem for cryptocurrencies [25].

Figure 1 shows an example account hierarchy where we label nodes with their roles (e.g., a MUA node has the M, U, and A roles). The initial node created by the genesis transaction is at the bottom. Each node is labeled with its set of roles. EachUA node represents an organization authorized to manage user accounts. The MUA nodes authorize the UA nodes and can undo any undesired action taken by the UA nodes, since theyare on the path from allUAnodes to the root. This action could be taken if there is negligence on the part of aUA nodein creatingU nodes orifaUA node’s credentials are stolen. Note that there are two MUAnodes, one on top of the other. The topmost node will be used to create and delete UA nodes, the bottom one will be used to fx the system in the event that the topmost node’s credentials are stolen. This is also the reason whythere are two MCLUA nodes, one on top of the other. The root node ideally is never used again after creating the MCLUA node above it. This helps prevent the root node’s credentials from being stolen. In general, actions should be performed by nodes higher up in the tree that have the least privilege possible since the use of a node puts it in a more vulnerable position. The credentials of nodes not used can be secured simply by converting them to physical form and locking them in a safe (which we recommend doing with the initial node’s credentials). This hierarchical node and role structure then enables the currencyadministrator to create a defense in depth security model. Accounts lower in the hierarchy have greater power and their credentials should be locked securely and rarely used.

A last capability not yet discussed is that accounts withM roles can issue policy that alters the cryptocurrency specifcation. In the event of policy conficts between different M nodes, the nodes closer to the root aremore authoritative.For M nodes the same distance from the root, those labeled with theMrolein earlier blocks are more authoritative.In theevent

Fig. 1. Example Managed Cryptocurrency Hierarchy.

ofa tie, the node labeled with theMrole frst within the same block wins.

The policy deployed by the M nodes defne the cryptocurrency. It is this policythat makes our approach an architecture. The policy can be set such that the cryptocurrency acts in an entirely unmanaged mode like the many popular open consensus cryptocurrencies in use today. The policy can also be set to allow the currency administrator full control as with the administrators in Multichain. More interesting to our research though is when the policy combines both open consensus and managed currencyfeatures. The policyenables each of the roles to be enabled or disabled and grants/limits the power of each role. Policy also can affect the mining community. A policy transaction can set a particular block reward or defne a minimum transaction fee. Controlling these will affect the size of the mining community. For a proof-ofwork based consensus mechanism such as Bitcoin, this will then indirectly control the amount of electricity used to manage the cryptocurrency (trading off power consumed against robustness of the mining pool against attack). This approach can enable an energy effcient proof-of-work consensus system where the currency administrator balances overall mining power desired vs. energy consumed. The exact capabilities available with policy are covered in section V-C.

IV. BITCOIN SPECIFICATION OVERVIEW

There does not exist an offcial Bitcoin specifcation. The original Bitcoin paper [4] contained the primary architectural detailsbut the specifcationis defnedby the applications that maintain it on the network. That said, there exists a Bitcoin reference client ’bitcoind’ and related protocol documentation [26]. From this was created a useful developers reference [27]. An in depth research analysis of Bitcoin is available in [28].

In this section we briefy review the features of the Bitcoin specifcation that will be of use for our modifed specifcation. Figure 2 shows the layout of a Bitcoin transaction (copied from [27], see this for details). The vin[] sections describe the inputs to a Bitcoin transaction (the particular coins to be spent). The hash and n values specify particular coins from the output of some other Bitcoin transaction. The scriptSig is a script to provide cryptographic evidence that the owner of the coins approves of the coins being spent. It is a response