BBc-1 : Beyond Blockchain One

- An Architecture for Promise-Fixation Device in the Air -

Kenji Saito and Takeshi Kubo
{ks91|t-kubo}@beyond-blockchain.org

Revision 0.1 – October 31, 2017

1 Introduction

Blockchain technology today has some sustainability risks including cryptographic techniques being used becoming obsolete and the price of native tokens declining (thus miners get incentivized to leave). There are also other technological problems and problems of technical governance, involving political situations surrounding development communities and miners.

BBc-1 (Beyond Blockchain One) is a kind of "middleware" to give long term solutions to these problems, and at the same time to provide support for currently ongoing application development. It is designed to implement requirements specified in "BBc Trust"[1], and is being developed as a set of open protocols and open source software that implement them, obtainable and usable for free[2]. We are in the process of forming a global open community, a non-profit consortium of corporations and individuals who will keep working on this project together.

BBc-1 provides proofs of transactions to applications. Initially, it is done by utilizing proof-of-existence functionality of existing blockchains. A blockchain to be used for this purpose can be dynamically chosen.

Today, major applications of blockchains can be categorized into two: currencies (systems for transferring quantities) and assets (systems for provenance and managing rights). BBc-1 will support both, and provide API and SDK for these.

BBc-1 will work as a collection of nodes being deployed and autonomously cooperating over virtually maintenance-free environments, minimizing administration cost.

The abstraction layer for blockchains will evolve so that it will work without underlying public ledgers in the near future. In the end, we aim to replace blockchains themselves with BBc-1.

Figure 1 shows an overview of the BBc-1 architecture.

[1] Our charter. Found at https://beyond-blockchain.org/public/bbc-trust.pdf

[2] GitHub repository is found at https://github.com/beyond-blockchain/bbc1

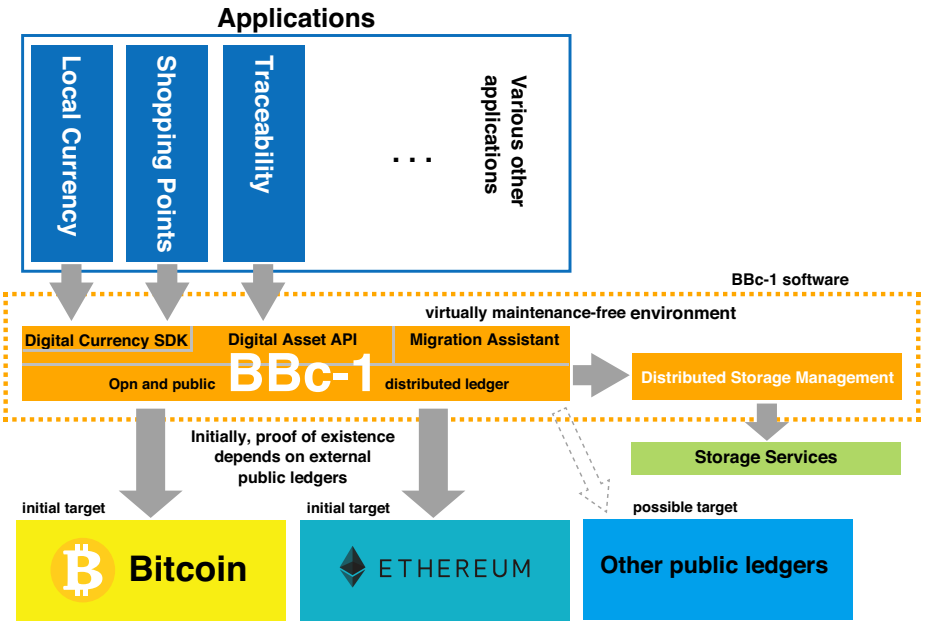


Figure 1: BBc-1 Architecture

2 Understanding Blockchains

What blockchains or other kinds of distributed ledgers do can be understood in terms of functional layers as follows (in the lower-layer-first order):

1. Guarantee of Validity (of transactions)
    - The ledger guarantees that a new transaction cannot be mutated, does not contradict with the existing history of transactions, and is committed by a user or users who have right to do so.
    - The ledger also guarantees that nobody can stop a user or users who have right to do so to commit a new valid transaction.
2. Proof of Existence (of transactions)
    - The ledger provides the proof of existence of a transaction.
    - The ledger does not allow anyone to delete the evidence of a transaction committed in the past, or to fabricate an evidence of a transaction that has never been committed in the past.
3. Consensus on Uniqueness (of transactions)

- If two contradicting transactions were to be committed, all users of the ledger (will eventually) see the same one of them in their views of the correct history of transactions.

4. Descriptions of Rules (that define semantics of transactions)

    - The ledger allows users to define semantics of transactions. (In Bitcoin, all transactions are basically about sending bitcoins.)

By providing these functions, a blockchain or a distributed ledger can act as a "promise-fixation device in the air", in which nobody can deny the content or existence of committed promises or records whose validity anyone can verify, and nobody can stop legitimate users to commit or verify such promises or records.

We will describe features of BBc-1 in terms of each layer above.

3 Features

3.1 Guarantee of Validity

BBc-1 uses a data structure similar to so-called UTXO (Unspent Transaction Output) structure used in Bitcoin and alike, because of its ease of understanding and of accounting.

To support confidentiality of transactions, BBc-1 works over inter-connected multiple domains of networks, where the content of transactions, including identities of involved parties, is visible only within each domain (digital assets handled in transactions can be encrypted to further support confidentiality).

Figure2 shows the structure of a BBc-1 transaction data, where

Header section contains meta-data of the transaction such as timestamp,

Events section contains set of operations over assets as outputs of the transaction, and also specifies the parties who can perform further operations over the assets,

References section contains references to past events and indications of <signature, public key> pairs in the signatures section as the proof of identities,

Cross-Ref section contains the identifiers of transactions from different domains that this transaction provides proof of existence, and

Signatures section contains a set of <signature, public key> pairs that sign the identifier of this transaction.

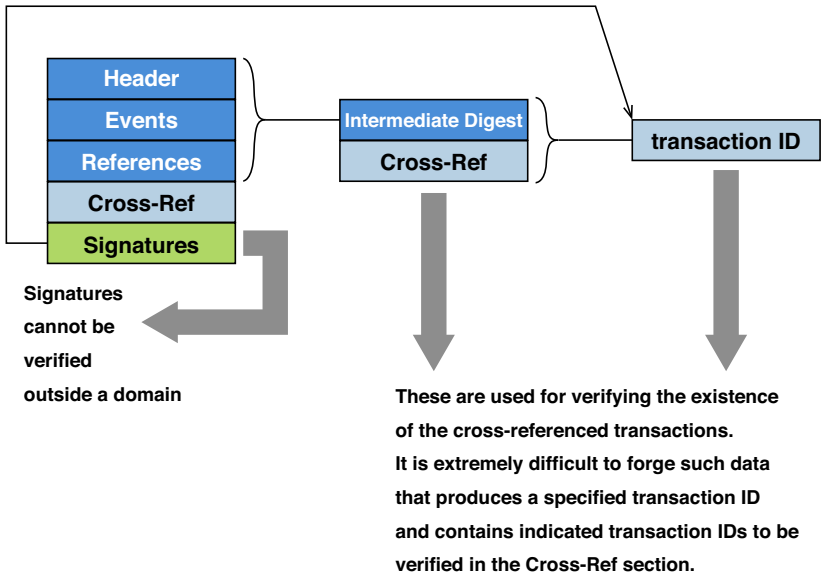


Figure 2: Structure of a BBc-1 Transaction Data

Blockchains today generally use the digest of a public key as an identifier of an external actor. This means that if the corresponding private key is lost, proving the identity of the user by digital signature will no longer be possible, and the user loses control of coins or assets forever.

BBc-1 advances solutions to this problem by separating identifiers and sets of public keys as shown in Figure 3. The binding between an identifier and a set of public keys is stored in the ledger itself (therefore, the binding is visible only within the domain, and the signatures cannot be verified outside the domain).

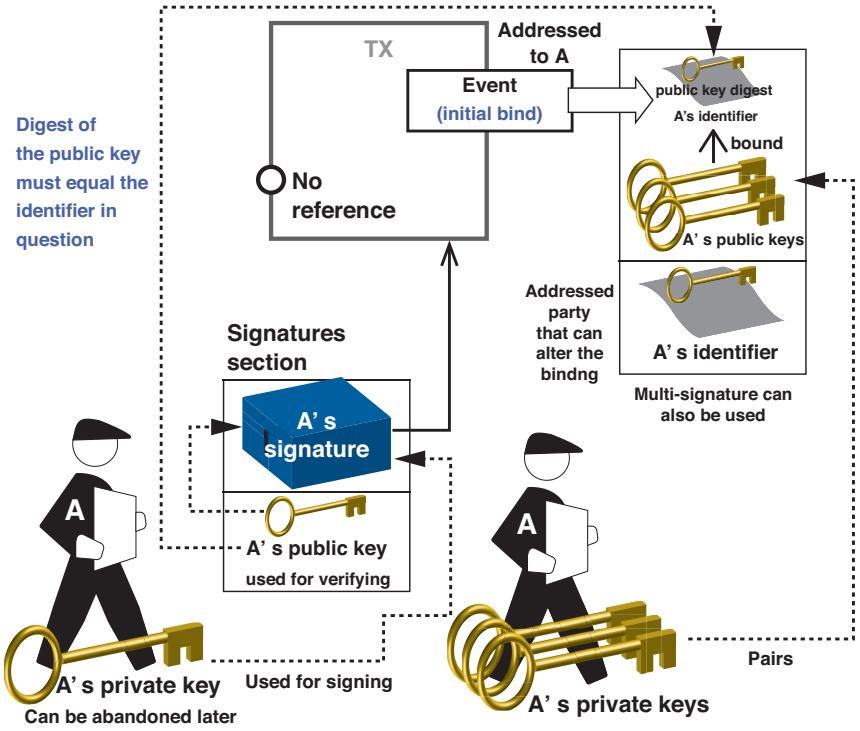
When verifying a digital signature, a verifier sees whether the specified public key is bound to the addressed identifier on the ledger or not.

3.2 Proof of Existence

At least in the initial stage of development of BBc-1, a proof of existence of a transaction is performed by checking the root of a Merkle tree written onto a public blockchain as illustrated in Figure 4, as commonly practiced in blockchain applications.

However, because a digest can be calculated by anyone, this method opens up possibility of attacking to general public. Therefore, we will eventually need to adopt a different method than using hash trees (including Merkle trees) or hash chains.

Instead of these techniques, we will use a cross-reference method in which a transaction provides proof of existence to a set of non-related past transac-



* An identifier is the digest of a public key, whose pairing private key must be used for signing the TX that initially binds the identifier and public keys.

Figure 3: Separation of Identifier and Public Keys

tions by containing their identifiers in the cross-ref section of the transaction data (Figure 5). This type of technique is often referred to as *DAG*, because the transactions form a directed acyclic graph of reference relations. In BBc-1, these cross-references are performed among transactions in unrelated domains, to minimize possibilities of colluding to alter records.

Moreover, by having timestamp services trusted and chosen by each user periodically committing time-defining transactions, general transactions can be proven to have existed in some bounded real-time in the past (again, Figure 5).

3.3 Consensus on Uniqueness

It has been well known by now that blockchains do not actually achieve consensus, through some enlightenment efforts or real cases and risks of hard-forking events.

This problem, we believe, is due to a design fault in which possibility of consensus among undefined participants is pursued, which we believe is

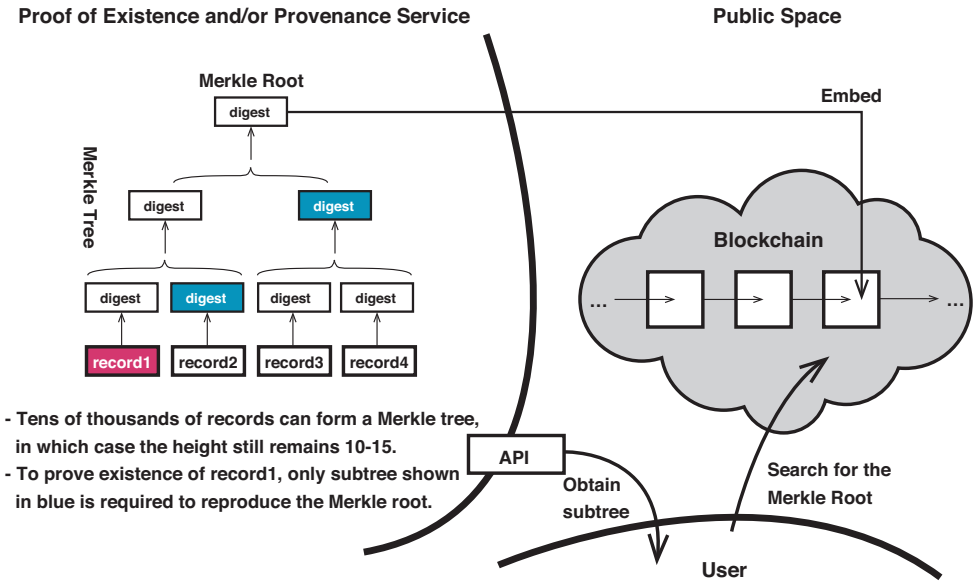


Figure 4: Proof of Existence Scheme using a Blockchain

impossible.

In Bitcoin, for example, coins are purely assets, without being backed by any debts. This means that when a coin is double-spent (thus the value is copied and doubled), it is unclear who is at a disadvantage. This in turn requires such a design that the whole participants as a group must agree on the uniqueness of transactions, which we know is impossible because we cannot even define "the whole" in an open environment.

On the other hand, if a coin or an asset is a representation of some debt, and there is a specific debtor, then it is clear where the motivation lies to become responsible for assuring the uniqueness of transactions.

Based on this thought, in BBc-1, addressing of payment or transfer of rights is generally specified with multiple signature requests, where a transaction requires a signature of the debtor in addition to that of the current holder to get committed. This might mean that the debtor is a single point of failure, but we believe that it can be handled by existing techniques of redundancy and consensus. In BBc-1, because identifiers and public keys are separated, the requests to debtors for signing can be anycasts (any signature will do as long as it is verified with a public key bound to the identifier in question). This means that each redundant node can have its own signing private key, and the first one to receive the request and to sign can be the proposer in the consensus protocol to be used, such as (Byzantine) Paxos.

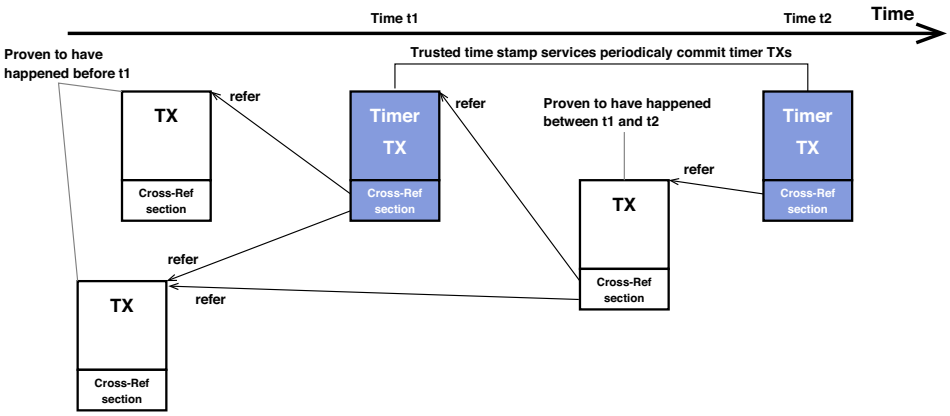


Figure 5: Proof of Existence of Transactions and its Time

3.4 Descriptions of Rules

In the future, BBc-1 will allow users to define smart contracts with languages designed for that purpose, but the mechanism for it is now in the process of being designed.

At least in the initial stage of development of BBc-1, applications will have to include their own logic to define and interpret the semantics of transactions they handle. We provide an SDK for the purpose.

3.5 Networking

Detailed descriptions of our intra-domain and inter-domain networking will appear.

4 Design Tasks

The following is the list of design tasks for us to work on:

- How transactions are committed by multiple signing actors representing a single user. This is a normal consensus problem that should be solved by some (Byzantine) Paxos protocol, but we need to develop a standard way of doing so for BBc-1 transactions.
- An efficient way to confirm the proof of existence using cross-reference technique.
- Mechanism and languages for smart contracts.
- How to interface with existing storage services.

  - It is thought beneficial to use external storage service for ease of deployment and guaranteeing the data storage.

- Incentives for nodes to join (probably better not to depend on issuing coins; but rather a tit-for-tat mechanism that disallows BBc-1 services to those who do not really participate in the network).

Revision History

Rev.0.1	2017-10-31	Initial revision

— End of Document —