BTCR v0.1 Decisions

a white paper from Rebooting the Web of Trust VII

by Kim Hamilton Duffy, Christopher Allen, and Dan Pape
with Ryan Grant, Anthony Ronning, Ganesh Annan, Wolf McNally

ABSTRACT

The Bitcoin Reference (BTCR) DID method supports DIDs using the Bitcoin blockchain. This method has been under development through Rebooting Web of Trust events and hackathons over the past year. The BTCR method's reliance on the Bitcoin blockchain presents both advantages and design challenges. During RWOT7, the authors made a number of design and implementation decisions -- largely scope-cutting in nature -- in order to lock down a Minimum Viable Product (MVP) version, which we'll refer to as v0.1. This paper documents those decisions, which will apply to the upcoming v0.1 BTCR method specification and associated v0.1 BTCR reference implementation.

OVERVIEW

The design decisions include:

- What's in and out of scope
- BTCR semantics
- Wallet MVP requirements and functionality
- Credential schema and content

SCOPE CLARIFICATIONS

To reduce design and implementation complexity, we decided on some reductions for a 0.1 release of BTCR. A variety of factors influenced these decisions, including lack of library support for our MVP scenarios and our desire for simplicity. Later versions of BTCR will address more advanced scenarios, including cost reduction, improved library security, and, most significantly, support for the mainnet chain.

1. Assume P2PKH scripts

Our initial BTCR prototypes relied on Pay-to-Public-Key-Hash (P2PKH) scripts. We explored other script types, but these introduced complications for extracting the signing public key directly from the transaction, which we require for the final DID document.

For this reason, we decided that only P2PKH scripts are supported, and other scripts types -- for example P2SH and Segwit -- are out of scope.

Because of the size and cost implications, we will revisit this decision in later versions.

2. Only an HTTP URL in OP_RETURN (Note: no IPFS support)

Background/Context

A BTCR transaction allows an optional OP_RETURN field pointing to a "continuation" DID document, which is a DID document containing additional key material and capabilities to be merged into the final BTCR DID Document (generated by the resolver).

The storage type of the continuation document introduced a fork during DID document resolution: if the link in the transaction pointed to mutable storage, the document could be updated after the transaction with the known DID (which isn't known until after the tx has been confirmed). However, if the link was a cryptographic hash link, then the document could not be changed without invalidating the hash.

Our original documentation described two paths to address these scenarios: - If the document is in an immutable store, we consider the transaction signature an implicit signature on the immutable content. - Otherwise, require a signature on the continuation DID document.

However, this introduces a requirement for the resolvers to be aware of different link types (which are content hashes or not), which we've tracked here.

v0.1 Scope Reduction: no special path for immutable continuation DID documents

The BTCR team strongly prefers use of immutable storage for its simplicity; however, we considered the burden of ensuring availability of IPFS objects prohibitive for evaluting BTCR DIDs in end-to-end scenarios. This would require the user to run an IPFS node (or have their objects pinned on one), and this isn't feasible on a mobile

device. For example, on an iPhone (our MVP target device) the IPFS lib on iPhone has to be running all the time, and can't be running in the background.

For these reasons, we decided to wait for greater support for IPFS on iPhone or to use something like filecoin. In v0.1 we will assume that, if the OP_RETURN is present, it points to an HTTP URL. We also assume that the target content could have been altered.

This allows us to cut scope and address the case of mutable storage only.

3. Txrefs and Txref-ext

The finalization of the txref spec (BIP-0136) is currently still in progress, but we have decided that BTCR DIDs will use the extended form of the txref, which encodes the TXO index within the transaction, as well as the block height and TX index of the transaction itself. See here for details.

4. Continuation DID Document in github (if the tx has an OP_RETURN)

Extending on the previous decision, we've decided in v0.1 to store continuation DID Documents in github. The continuation DID Document must be updated after tx confirmation to explicity list fields we formerly derived as part of the implicit DID document.

Specifically, after tx confirmation, the user must specify: 1. the resulting DID in relevant fields such as `id` and `creator`. Note this must be done after confirmation because the DID will be known only after the BTCR tx is confirmed. 2. a signature on the updated DID Document from the tx signing key

The decision to store it in github specifically reduces the target audience, since it requires a github account. However, we were already assuming a dev audience for v0.1.

One advantage is of this restriction is the ability to sign DID Document updates with a PGP key (via github signed commits). The flow would look like this:

1. Create BTCR DID
2. Create the full BTCR DID document in github, including the PGP key in this document
   ⇨ Note the DID document itself is signed with BTCR transaction signing key, as usual
3. Commit to github using PGP key from (2).

5. Testnet only

Perhaps most notably, we've decided to only support testnet (not mainnet) as we work through the initial reference implementations and obtain feedback from test usage.

SEMANTICS

A BTCR DID document relies partially on transaction structure, partially on the continuation DID document. We clarified the semantics of scenarios that were previously undefined.

1. Transaction input/output semantics

a. TX Input addresses

In v0.1, the keypair corresponding to the first TX input has the following properties: - it is the only key that can be used to verify control of the DID and continuation DID document - it must be used to sign the updated continuation DID document (after TX confirmation)

If an OP_RETURN doesn't exist, the keypair corresponding to the first TX input is granted the following additional capabilities (by the resolver): - DID auth - Sign/verify Verifiable Credentials

If the OP_RETURN exists, the continuation DID document obviates the need for additional implicit capabilities; we assume each capability is listed explicitly in the continuation DID document.

b. TX Output address (1st monetary output)

The output address is used for "following the tip". If spent, the BTCR DID has either been rotated or revoked. Resolvers must follow the tip to find the latest transaction.

Determining whether the DID has been rotated or revoked is described in the next section.

2. No OP_RETURN after a TX means revoked

A BTCR DID is considered revoked if: - The latest transaction has no OP_RETURN, AND - There is more than one transaction in the BTCR DID "chain"

The first factor is important because a missing OP_RETURN is considered valid in the very first TX in the chain. However, all subsequent TXs in the chain must have OP_RETURNs, or else it is considered revoked.

This behavior must be enforced by BTCR resolvers.

WALLET REQUIREMENTS AND FUNCTIONALITY

We will release a v0.1 BTCR wallet as an iPhone app. We wanted to include sufficient features to demonstrate core BTCR features (as described in the rest of this paper). While defining requirements, we included some usability features and also realized library limitations affecting our implementation choices.

1. Import previous Bitcoin transactions

The v0.1 BTCR wallet implementation will support importing existing private keys corresponding to unspent outputs from previous transactions. This allows users to generate a valid BTCR DID from an existing transaction, with no additional cost.

A consequence is that a transaction doesn't need to be created or broadcast (from within the ID wallet) to instantiate this initial BTCR DID. The BTCR wallet only needs to create transactions for any subsequent update/revoke operations.

In this case, the BTCR DID will have no continuation DID Document.

2. Updates must use non-financial HD derivation paths

We want to allow users to share mnemonic seeds across their ID wallet and their Bitcoin wallets. To achieve this, we must make sure addresses backing valid DIDs are not accidentally spent, which would result in a BTCR DID revocation.

We will achieve this by choosing our own convention about the derivation path.

3. Tip following

Many current BTCR prototypes (such as the BTCR playground) use Bitcoin APIs to look up transactions. This introduces many concerns from security to availability (rate-limiting).

We want our MVP deployment to require minimal resource overhead. To achieve this, our preference is to follow the best practice described in BIP 157/158 (aka Neutrino), which helps preserve the privacy of your DID-related addresses (over SPV).

> *Note that Neutrino is not as efficient as SPV. SPV monitors only the addresses it cares about. In contrast, Neutrino introduces noise in attempt to hide the specific addresses it actually cares about.*

The problem we encountered is that library support for Neutrino is not readily available.

As a fallback, we will use REST services backed by a bitcoin node (initially maintained by the BTCR team).

4. Pre-revocation

The v0.1 BTCR wallet may support pre-revocation in cases of emergency. This works by pre-signing a transaction spending the tip, which can be stored away and broadcast later to revoke the DID even without the key material.

CREDENTIAL SCHEMA AND CONTENT

This section applies to credential schema and content, which is relevant when demonstrating DID use cases. Consistent with the previous decisions, we urge people to use common schemas (for ease of interop) and not to issue high-stakes claims (for privacy/security).

1. Use JSON-LD 1.1 javascript lib, because 0.1 doesn't need high-stakes verifiers

As many of the decisions above imply, BTCR v0.1 is not recommended to be used for anything other than experimentation. In a future production-ready version, we will use more secure libraries (and generally avoid reliance on javascript, as we've done for prototyping).

This implies that where JSON-LD libraries are required in BTCR v0.1, we may use the JSON-LD 1.1 javascript library.

In future versions, we will require library support in a different language.

2. Restrict to schema.org schemas, like person.

We'll restrict our prototypes to schema.org schemas as opposed to custom schemas. We've developed a suite of test cases that can be used.

3. Privacy: stick to pseudoanonymous claim content

Users should not include any PII in claims used for BTCR v0.1 prototypes

4. Restrict to simple VCs one wishes to share

Similar to above, only include claim content that you want others to see. This phase is not equipped to handle high-stakes scenarios.

EXAMPLE

```
{
  "@context": "https://w3id.org/did/v1",
  "id": "did:btcr:8kyt-fzzq-qpqq-ljsc-5l",
  "publicKey": [
    {
      "id": "did:btcr:8kyt-fzzq-qpqq-ljsc-5l#keys-1",
      "owner": "did:btcr:8kyt-fzzq-qpqq-ljsc-5l",
      "type": "EdDsaSAPublicKeySecp256k1",
      "publicKeyHex":
"0280e0b456b9e97eecb8028215664c5b99ffa79628b60798edd9d562c6db1e4f85"
    },
    {
      "id": "did:btcr:8kyt-fzzq-qpqq-ljsc-5l#keys-2",
```

```
      "type": "RsaVerificationKey2018",
      "publicKeyPem": "-----BEGIN PUBLIC KEY...END PUBLIC KEY-----\r\n",
      "owner": "did:btcr:8kyt-fzzq-qpqq-ljsc-5l"
    }
  ],
  "authentication": [
    {
      "type": "EdDsaSAPublicKeySecp256k1Authentication",
      "publicKey": "#keys-1"
    },
    {
      "type": "RsaSignatureAuthentication2018",
      "publicKey": "#keys-2"
    }
  ],
  "service": [
    {
      "type": "BTCREndpoint",
      "serviceEndpoint":
"https://raw.githubusercontent.com/kimdhamilton/did/master/ddo.jsonld"
    }
  ],
  "SatoshiAuditTrail": [
    {
      "chain": "testnet",
      "blockhash":
"0000000000000722ded9d85d67e145ba41c53ef2e8680f75540a08b885febba5",
      "blockindex": 2,
      "outputindex": 1,
      "blocktime": "2017-09-23T17:27:56.682Z",
      "time": 1499501000,
      "timereceived": "2017-09-23T17:27:56.682Z",
      "burn-fee": -0.05
    }
  ],
  "claims": [

  ]
}
```

ADDITIONAL CREDITS

Lead Author: Kim Hamilton Duffy

Authors: Christopher Allen, Dan Pape

Contributors: Ryan Grant, Anthony Ronning, Ganesh Annan, Wolf McNally

---

About Rebooting the Web of Trust

This paper was produced as part of the Rebooting the Web of Trust VII design workshop. On September 26th through 28th, 2018, over 40 tech visionaries came together in Mississauga, Ontario to talk about the future of decentralized trust on the internet with the goal of writing 3-5 white papers and specs. This is one of them.

Leadership Team: Christopher Allen, Joe Andrieu, Kim Hamilton Duffy, Manu Sporny, and Heather Vescent

Gold Sponsors: Civic, Protocol Labs, Sovrin, an anonymous donor

Silver Sponsors: HTC, Microsoft, PTB Ventures, Spherity, Tierion

Community Sponsors: Blockchain Commons, Learning Machine, Legendary Requirements, Purple Tornado, Veres One

Workshop Credits: Christopher Allen (Founder), Joe Andrieu (Producer and Facilitator), Shannon Appelcline (Editor-in-chief), and Claire Rumore (Graphical Recorder)

Thanks to our other contributors and sponsors!

What's Next?

The design workshop and this paper are just starting points for Rebooting the Web of Trust. If you have any comments, thoughts, or expansions on this paper, please post them to our GitHub issues page:

https://github.com/WebOfTrustInfo/rwot7/issues

The next Rebooting the Web of Trust design workshop is scheduled for the week of March 1st to March 3rd in Barcelona, Spain. If you'd like to be involved or would like to help sponsor the event, email:

rwot-leadership@googlegroups.com
