Indestructible Data: How Arbitrary Data is Added to the Blockchain

Author	josh
Date	2020-03-13 18:06:04

Overview

The blockchain is a very powerful tool - it's a distributed, immutable database of all the transactions that have ever occurred on a cryptocurrency network. When you transact on Bitcoin, for example, that transfer of value is recorded on the blockchain *forever*. It cannot be modified or changed after just a few confirmations (blocks added on top of that block) from miners.

But it's not just raw transaction data that can be added to blockchains. Many of the most popular allow *arbitrary* data to be added to chains. We'll talk a bit about how data is added to Bitcoin and Bitcoin forks in particular.

Adding to the Database

Understanding Transaction Script Basics

When you transfer value from one party to another on the chain, there's not one hard-coded transaction type. In reality, Bitcoin transactions are *programmable*, allowing the use of a small scripting language appropriately called *Script*. Here's what you need to know about Bitcoin scripts: the end goal is to end up with a value of *true* on the Script stack, after the script finishes execution. If the end value of the program is true, then the transaction is considered valid.

For example, a typical script looks like this:

```
<Signature> <Public Key> OP_DUP OP_HASH160 <Public Key Hash> OP_EQUALVERIFY
OP_CHECKSIG
```

This is called a "Pay to Public Key" transaction, and here the script evaluates to *true* as long as the spender provides a valid signature from their private key. This script is built from the original *locking script* (OP_DUP onward) on the UTXO at the spender's address , concatenated to the *unlocking script* (sig and pubkey) created by the spender.

Data Scripts with OP_RETURN

So what does that have to do with adding data to the blockchain? Well, it turns out scripts have a great available operator called `OP_RETURN`. This operator simply marks the transaction as provably unspendable immediately! It is still considered a valid transaction.

And after the OP_RETURN, there is room leftover for arbitrary data. So when the transaction is included in the chain, the OP_RETURN allows the transaction to be valid, and the other data gets recorded on the blockchain!

For example, I could create a transaction with the data:

```
OP_RETURN chaintuts teaches bitcoin!
```

to forever record on the blockchain my enjoyment of teaching and explaining.

Now the amount of data one can store is not unlimited, for economic and scaling reasons. It would not be feasible for the chain to store users' files and that sort of thing. For this reason Bitcoin BTC limits OP_RETURN transaction data to a size of 40 bytes. BCH allows 220 bytes, and BSV allows 100,000 bytes.

OP_RETURN Data Forever, and Cool Use Cases

Now it might be fun to record my love of teaching on chain, but it's not really that useful. The amazing thing about OP_RETURN is the ability to built decentralized applications on top of Bitcoin. The Bitcoin Cash chain has applications like memo.cash, where users can post on a "decentralized Twitter" hosted on the chain. BCH also uses OP_RETURN scripts for tokenization, using the SLP standard.

Although the amount of data is limited for economic and scaling reasons, there are tons of possibilities for this type of decentralized data storage, so long as it is used within reason. Imagine the Bitcoin apps that can be built!