With the upcoming hard fork in BCH in mid-November, we wanted to explain the main differences between Soft and Hard forks in the cryptocurrency space.

A *fork* in programming terms, is an open-source code modification. Usually the forked code is similar to the original, but with important modifications, and the two *prongs* comfortably co-exist. Sometimes a fork is used to test a process, but with cryptocurrencies, it is more often used to implement a fundamental change, or to create a new asset with similar characteristics as the original. The majority of cryptocurrency forks are due to disagreements over embedded characteristics. Forks have a "shared history." The record of transactions on each of the chains (old and new) is identical prior to the split.

A hard fork is a change to a protocol that renders older versions invalid. If older versions continue running, they will end up with a different protocol and with different data than the newer version. This can lead to significant confusion and possible error.

With bitcoin, a hard fork would be necessary to change defining parameters such as the block size, the difficulty of the cryptographic puzzle that needs to be solved, limits to additional information that can be added, etc. A change to any of these rules would cause blocks to be accepted by the new protocol but rejected by older versions and could lead to serious problems – possibly even a loss of funds.

As an example, if the block size limit were to be increased from 1MB to 4MB, a 2MB block would be accepted by nodes running the new version, but rejected by nodes running the older version.

This is a hard fork, and it's potentially messy. It's also risky, as it's possible that bitcoins spent in a new block could then be spent again on an old block (since merchants, wallets and users running the previous code would not detect the spending on the new code, which they deem invalid).

The only solution is for one branch to be abandoned in favor of the other, which involves some miners losing out (the transactions themselves would not be lost, they'd just be re-allocated). Or, all nodes would need to switch to the newer version at the same time, which is difficult to achieve in a decentralized, widely spread system.

A soft fork can still work with older versions.

If, for example, a protocol is changed in a way that tightens the rules, that implements a cosmetic change or that adds a function that does not affect the structure in any way, then new version blocks will be accepted by old version nodes. Not the other way around, though: the newer, "tighter" version would reject old version blocks.

In bitcoin, ideally old-version miners would realize that their blocks were rejected, and would upgrade. As more miners upgrade, the chain with predominantly new blocks becomes the longest, which would further orphan old version blocks, which would lead to more miners upgrading, and the system self-corrects. Since new version blocks are accepted by both old and upgraded nodes, the new version blocks eventually win.

This is a soft fork, and it's already happened several times. Initially, Bitcoin didn't have a block size limit. Introducing the limit of 1MB was done through a soft fork, since the new rule was *stricter* than the old one. The pay-to-script-hash function, which enhances the code without changing the structure, was also successfully added through a soft fork. This type of amendment generally requires only the majority of miners to upgrade, which makes it more feasible and less disruptive.

Soft forks do not carry the double-spend risk that plagues hard forks, since merchants and users running old nodes will read both new and old version blocks.