Mining: An Overview

We can break the process of mining down into six simple steps:

1. Join the network and listen for transactions.
2. Listen for new block sand maintain the block chain.
3. Assemble a new valid block.
4. Find the nonce to make your block valid.
5. Hope everybody accepts your block.
6. Profit!

Miner Misbehavior

How can miners deviate from the default protocol?

- **Include invalid transactions in a block**. This is relatively easy to shoot down: everybody else would reject the block.
- **Withhold blocks that they've mined**.
- **Blacklist certain addresses or transaction types**. This still produces valid blocks. They could blacklist either by excluding transactions that they don't like or try to fork when others send them blocks containing such transactions.
- **Not relay certain (or any) blocks**.
- **Spam or DDOS the network**. This can be done by propagating invalid blocks or even creating invalid blocks.
- **Build off of an old block**.
- **Blacklist a miner or pool** by never building off of their blocks.
- **Try to fork away from undesired blocks**, i.e., those created by blacklisted miners.
- **Scoop-up "dead money"**, e.g., anyone-can-claim transactions, or transactions with weak passwords.

- **Break ties differently**. Maybe you always build off of the *second* block you hear about, or accept bribes from other miners to build off of their blocks (to prove that you're mining on top of the bribers block, you could send them shares (see the notes on "Mining Pools" for more)), or even accept the block with lower transaction fees so that you can put the high-fee transactions in your own blocks.
- **Change timestamps**. There are some limits to what you can put in that field, but you have a bit of leeway. For example, by making it look like blocks are coming out slowly (i.e., increasing the timestamp), you can prevent the difficulty from increasing too quickly.
- Out-of-band: **Coerce other miners**.
- Out-of-band: **Collude or share mining capacity** ("laundering" hashes or blocks).

Feather Forking

This is an attack first conceived by Andrew Miller. Say a block X that you don't like is propagated to you. Assume X is preceded by block Y. The goal is to try and build two blocks, built on top of Y, so that your branch becomes the longest before the network really builds upon X. (If two blocks quickly come in and build on top of X, then you give up.) The goal is to have the rest of the network follow you instead of block X.

An Empirical Formula

Define α to be the attacker's proportion of the network hash capacity, and define k to be the number of blocks behind that you're willing to continue attempting the attack (i.e., your persistence).

Set $k = 2$, for simplification. There's an α^2 chance that you find two blocks before they find any, and there's an $\alpha(1-\alpha)$ chance that you end up in the exact same scenario as you were in at the beginning (i.e., that you find a block, then the main branch finds a block). This gives us the following recursive relationship: $E = \alpha^2 + \alpha(1-\alpha)E$. Solving gives us $E = \frac{\alpha^2}{1-\alpha+\alpha^2}$.

If you can get E up to 20% or so, then there's not really any reason for others to include X in their versions of the blockchain. Even though you will probably lose money by executing the attack, there's enough of a threat (if you guarantee to go through with it) that you can successfully blacklist the miner of block X.

This attack relies on visibility: you need to make your intentions public to get the other miners to go along with it.

Comparisons to the 51% Attack

In the original Bitcoin whitepaper, Satoshi discusses the 51% attack, in which one miner uses his or her majority mining share to overtake the main branch and double spend. The author

argues that this attack would never happen, as it would be against the attacker's economic interest: if a 51% attack is launched, then everyone would lose faith in the currency and it would lose all of its value, making the attack worthless to the attacker.

The 51% attack is clearly an attack, but you could probably find a reasonable political argument for launching a feather fork (e.g., you want to blacklist any transactions from the Silk Road).

The other major difference is that fighting this attack requires loyalty from the miners: they might get together and agree that it's in their collective benefit to fight the attacker (e.g., to include transactions from an address that the attacker has chosen to blacklist), but when they go off on their own, they don't *really* have an incentive to do so. This is a classic game theoretic problem known as the "Tragedy of the Commons".

Goldfinger Attack

A Goldfinger attack is one in which your value comes from some other resource burning. In this case, it might involve Bitcoin value rising by some other altcoin tanking; thus, a Bitcoin stakeholder might benefit from launching a 51% attack on the altcoin and watching it burn.

Selfish Mining

Another mining strategy: don't announce blocks right away; instead, hoard them to yourself in an attempt to get ahead of other miners. Typically, you'll try and mine two blocks in secret; then, even if someone else releases a new block, you can drop both of yours and everyone will switch to follow you. In the time before anyone finds a new block and after you've already found two, you have exclusive rights to mining the next block (because no one else in the network knows about it).

What if the network finds a block before you find your second (selfish) block? You have a few options:

- **Continue working on your own block**. There's a chance that you find a new one before they find another, and then everyone will switch to your chain. (Recall, though, that proof-of-work is memoryless, so it's not like the work you've put in to mining your secret block already counts for anything.)

- **Release your block and aim to have optimal network position**. You might be able to flood the network with your secret block before the initial block has been propagated. This strategy could be bolstered by maintaing a bunch of Sybils around the network–Sybils don't require any work (i.e., hash power) to set up; they can just sit around and do nothing until they need to relay your blocks (this also makes the attack hard to reason about).

How could we analyze this block-withholding attack? Assuming we have hash power α:

- With probability α^2, we find two blocks before anyone in the network finds a single block. We also have some time to mine the next block on our own, which is worth $\frac{1}{\alpha}$ to us.

- The attack backfires if we find a block, the network finds a block, we compete to propagate it onward, and lose. If we control the network (very unrealistic), then this attack would *never* fail. For any other proportion of network control, it's a race.

If you have a 50% chance of winning a race, then this is a profitable strategy for $\alpha > 0.25$. But this attack hasn't been observed in practice and is difficult to reason about–how can we compute the true probability of winning a race?