1. Hash functions
   Because the phone number is not long enough. In the extreme case, if we take a one-to-one correspondence between the input space and the output space, the size of the output space could be at most 10^{10}. This is definitely not a large possible space giving the current computing power of the commercial computer. Anyone who wants to maliciously determine the phone number just need to try hashing each possible input and compare with the target hash value. It will not take a lot of time to finally figure out the mapping between the phone number and the hash value.
2. Signatures
   The sign method takes a message and a secret key, sk, as input and outputs a signature for message under sk. The verify method takes a message, a signature, and a public key as input. It returns a boolean value isValid that will be true if sig is a valid signature for m essage under public key pk, and false otherwise. In order to operate over a smaller input, we can sign the hash of the message. If we use a cryptographic hash function with a 256-bit output, then we can effectively sign a message of any length as long as our signature scheme can sign 256-bit message. It's safe to use the hash of the message as a message digest in this manner since the hash function is collision resistant.
3. Proof-of-work
   The key idea behind proof-of-work is that we approximate the selection of a random node by instead selecting nodes in proportion to a resource that we hope that nobody can monopolize. In the case of Bitcoin, that resource is computing power. Bitcoin achieves proof-of-work using hash puzzles. Suppose instead of the existing proof-of-work, Bitcoin required n sequential hashes of a block to prove that n iterations of sha256 had been computed. It would be possible for the malicious node to try to forge the proof-of-work message. Say it can show to other nodes that it did n iterations but in fact just once. However, the hash puzzle could prevent this from happening. If the hash function satisfies the puzzle-friendliness property, then the only way to succeed in solving the hash puzzle is to just try enough nonces one by one until you get lucky.
4. Bitcoin
   This is because honest nodes' behavior is always to extend the longest valid branch that they see. The chance that the shorter branch with the double spend will catch up to the longer branch becomes increasingly tiny as it grows longer than any other branch. This is especially true if only a minority of the nodes are malicious — for a shorter branch to catch up, several malicious nodes would have to be picked in close succession. The transaction in the most recently broadcast block may reside in the shorter branch and get abandoned eventually, while the one in a block a few prior to the most recent block may have been extended for several times and the possibility it gets abandoned will go down exponentially.

   .