Why I find Iota deeply alarming

Nick Johnson

Follow

HackerNoon.com

--

51

Listen

Share

When I first learned about Iota, at Hack The Valley in January, my initial interest was quickly tempered by scepticism when I learned about some of its basic design choices — which we'll get into in a moment — and I largely ignored it as a hobbyist project writ large.

Its continuing popularity, and recent news surrounding its choice of hash function has caused me to think more deeply about why I find Iota alarming, and why that should matter.

My motivation here is to highlight what I see as critical technical and social issues with Iota, which I consistently see downplayed and ignored by many. I have no financial stake in the success or failure of Iota; if it seems that I have animus against it, it's purely because I believe good systems should succeed, and bad systems should fail, in each case on their own merits. In the interests of full disclosure, I am a core developer of Ethereum.

Iota shows a lack of good technical judgement

One of the first things you learn on investigating Iota further is that it usesbalanced ternary, a numeral system with 3 digits, -1, 0 and 1. The authors have various arguments as to why they made this decision, but they come down to two main ones:

- Ternary processors are theoretically more efficient than binary processors.
- Certain mathematical constructs are more cleanly represented in balanced ternary.

Unfortunately, neither of these are relevant in a practical system. Iota is by necessity built to run on existing hardware, which is exclusively binary, as are the communication networks it uses. As a result, all of its internal ternary notation has to be encapsulated in binary, resulting in significant storage and computational overhead. Math must either be performed on individual 'trits' or first converted from binary-wrapped-ternary encoding into the machine's native number representation, and back again afterwards — in either case imposing a large computational overhead.

Likewise, the theoretical benefits of a balanced ternary notation, such as not needing a sign bit, are more than outweighed by the practical disadvantages, since every processor Iota will run on is already equipped to perform math on twos-complement numbers, but requires software emulation to operate on balanced ternary.

This combination of not invented here syndrome and the Dunning-Kruger effect has led to a situation where the authors of Iota have decided that their affection for the tidyness of balanced ternary must outweigh all practical considerations in system design, and leads to a system that is needlessly complex.

Iota disregards cryptographic best-practices

Iota's novel choice of numeral system also requires them to reinvent basic operations such as cryptographic hashing. This violates rule 1 of cryptography: don't roll your own crypto. The predictable result of this has already been demonstrated, with a team led by Neha Narula demonstrating a number of significant vulnerabilities in Iota's hash function, Curl.

IOTA's authors argue that these flaws are not material to Curl's use in Iota, andthat collision resistance is not required, only preimage resistance. Personally, I don't find this persuasive; cryptographic attacks always get better, never worse, and the first collision attacks against MD5 and SHA1 were what prompted the cryptographic community to start moving off those functions, even for applications where only preimage resistance is required.

Iota is a bad actor in the open source community

Next, and in my mind most damningly, Sergey Ivancheglo, Iota's cofounder,claims that the flaws in the Curl hash function were in fact deliberate; that they were inserted as 'copy protection', to prevent copycat projects, and to allow the Iota team to compromise those projects if they sprang up.

It honestly astounds me that anyone would think this justification redeems them; it’s an admission of hostile intent towards the open-source community, akin to publishing a recipe but leaving out a critical step, rendering the resulting dish poisonous to anyone who eats it. It’s also very revealing of Iota’s attitude towards open source; they do not release code open source because they want to make it freely available to all and advance the state of the art, but rather because it serves their purposes in gaining adoption for their own deployment of that code.

If Iota wish to discourage copycats, they can license their code in a manner that prohibits the kinds of reuse they are unhappy with, or keep it closed source, as they have done with their centralised coordinator. That, of course, would lose them the approval of the open source community — but so should their actions here, in booby trapping the code they release.

Iota’s integrity guarantees lack rigor

Iota, rather than being constructed around a blockchain, utilises what they call a ‘tangle’ — a directed acyclic graph of transactions — with no single head. I’m enthusiastic about attempts to advance the state of the art in distributed ledgers, and I think this may have promise in terms of improving scale, but so far their arguments for the scalability of this system and its security seem very hand-wavy.

Each transaction is secured using a proof of work, but this PoW function has a fixed difficulty. Since Iota is designed to run on low-power nodes, the difficulty is quite low, and it would not take much in the way of dedicated resources to outweigh the entire processing power of the Iota tangle. Further, unlike blockchain-based systems such as Ethereum and Bitcoin, the difficulty of the proof of work is not adaptive. This means that the security of the tangle directly depends on the number of transactions being processed, and that there is no way to adapt the security level to real-world conditions. Ethereum and Bitcoin gain their game-theoretical soundness from the financial reward given to miners, and the guarantees this creates that an attacker must have more hash power at their disposal than all the honest actors combined. Iota lacks such a guarantee, and I’m unaware of any robust proof that Iota is secure against these sorts of 51% attacks.

Likewise, I’m unaware of any proofs that the tangle will always converge; that an attacker cannot force it to have an unbounded number of leading edges, or that the system remains secure when each node does not validate a majority of the transactions in the network. Iota authors have provided informal explanations of why they believe all of these to be true, but they fall far short of the level of assurance blockchain based systems are able to place in their proof-of-work (and soon, proof-of-stake) mechanisms.