NOTE: The views expressed here are my own and do not necessarily represent or reflect the views of others or my employer.

Thanks Albert Ni for review and discussion.

## Intro

Layer 2's importance to Ethereum is growing by the week, and everyone knows it.

However, "layer 2" is an imprecise label. Right now, when people say "layer 2", they tend to mean "not on Ethereum layer 1". But the way something interacts with Ethereum layer 1 matters a lot. Different solutions that are all considered "layer 2" can have wildly different properties. Arguably, layer 2 should only refer to certain things with certain properties (e.g. we probably all agree that something that lives on AWS is not layer 2, but there are projects that arguably have similar security guarantees which are considered layer 2). But maybe that's a topic for another day.

For this post I want to dig into the properties of sidechains.

Sidechain basically means a system where a set of validators checkpoint the latest state of the chain to a smart contract. These checkpoints are then used by a bridge contract to allow users to deposit and withdraw. There is usually a leader election process among the set of validators to determine who can create blocks. Examples of leader election mechanisms include proof-of-authority (POA) consensus, and Proof of Stake.

Sidechains have played an important role in the the ethereum ecosystem. They've been a stopgap solution for scalability and usability while the research community was still working on better solutions. Products like xDai at hackathons have highlighted the need for better UX which has trickled into the rest of the space.

However, sidechains do not have the security properties the broader Ethereum community expects

. That doesn't mean they should never be used. If people using sidechains are fully aware of the lack of certain properties, and still want to use them, that's their prerogative. It may be worth the tradeoffs. It gets dangerous when people are not aware. The hope of this post is to provide information. If everyone was already aware of these properties, then another post about it can't hurt. But if this helps people realize mistaken assumptions, that's a good thing.

What are the security properties sidechains lack? Almost all sidechains do not

provide:

- Censorship resistance
- Finality
- Guarantees about owning funds

If you want more of these properties, seeking alternatives to sidechain-based solutions is one path forward. It is also possible to improve on these dimensions while still using the core sidechain architecture.

My hope is that an open discussion of these properties will benefit everyone.

## Censorship resistance

It should not be controversial to say that sidechains have weaker censorship resistance properties than (well designed) blockchains. Otherwise, there would be no need for blockchains. However, let's break this down a bit further.

If N validators are involved in a sidechain, and a transaction can be censored as long as M validators agree to do so, then N-M is the number of validators that need to collude in order to censor a block. This leads to a tricky balance where making it harder to censor transactions makes it easier to censor blocks. Given that both transaction censorship and block censorship are undesirable, this makes it fundamentally difficult for sidechains to have strong censorship resistance properties.

This concern extends to when proof-of-stake is used. It potentially gets worse as the numbers involved are weighted by stake, which means the number of distinct entities required to reach certain thresholds is likely even lower (at best, stake is perfectly uniformly distributed, in which case it's just like the non-proof-of-stake case).

## Data Availability Guarantees

We know that N-M validators are able to create a block. We also know that all other validators need to have data about the whole state in order to validate the new state. So if N-M validators are malicious, they can

1. Create a new block
2. Refuse to share the data with the honest validators

3. Effectively remove N - (N-M) = M honest validators from consensus. Thus capturing the system.

How likely is this to happen? It obviously depends on many situation specific details, but we can start by considering what the incentives are for a rational validator to share data with all other validators. For traditional proof-of-authority, there is likely to be a reputational cost to not doing so. With proof-of-stake based sidechains, there may be stake at risk. However, it's not easy to make this work, because there is no way to prove that some data was left unavailable without someone else putting all the data on chain. If this sounds like optimistic rollup, it is, which means sidechains with better security properties essentially reduce to optimistic rollup.

In most sidechains, validators receive some form of payment for being validators. For honest validators, this reward is shared among N validators. For dishonest validators, this same reward is shared between N - (N-M) = M

(most importantly M < N

here), so the there is incentive for validators to not share updated state with others.

An overall concept to keep in mind is it is very difficult to diagnose data availability attacks. To honest nodes, they are often indistinguishable from sync problems.

## Finality

Imagine a series of state transitions as follows

state_1 => state_2 => state_3

Where each =>

involves a bunch of transactions being applied as part of updating state. Finality is the idea that once applied, a transaction cannot be undone.

Sidechains checkpoint blocks after they have been agreed upon via the consensus on Ethereum mainnet. This may lead one to think that sidechain finality basically is equivalent to Ethereum finality. Specifically, that in order to revert blocks on a sidechain, you would need to revert blocks on Ethereum. This is not the case

.

This is because Finality is about reverting transactions, not about replacing an old state with a new one. So N-M

validators are able to perform the following transition:

state_1 => state_2 => state_1

(replacing state_3

with state_1

, thus reverting the supposedly finalized state_2

without requiring Ethereum mainnet reversion).

## Guarantees around ownership of your funds on a sidechain

Assume there exists a state where state_1 = {\text{Alice}:1000, \text{Bob}:0}

So Alice has 1000, and Bob has 0. What happens if Bob is malicious and controls (or can effectively collude with) a super majority of POA validators?

Then, Bob can simply perform the state transition state_1 => state_2

where state_2 = {\text{Alice}:0 , \text{Bob}:1000}

Of course, this is tantamount to stealing all the funds from Alice and giving them to Bob.

Thus, a sidechain's defense reduces to saying N-M

validators could never be convinced to process such an illegal state transition.

This is well known (or so I believe), but I think it's useful to remind everyone how this works. Your confidence in a sidechain reduces to your confidence that a supermajority of a sidechain would never do something like this. Most analysis of a sidechain's security should focus on this.

Now, there may be groups of people (validators) that you would trust in this way. Just like many of us trust various centralized service providers for many things. Sometimes that's worth the trade offs. It's just important to be clear that that is the trade off being made.

## Issues with governance as defense

An argument is sometimes made that "we can just use governance to solve everything mentioned so far". This is flawed in that it basically says the whole system degrades to governance. One reason this argument especially concerns me is that it means the other attributes of the sidechain are theater (in which case, why have those attributes at all?). For instance, if governance is the final fallback to protect against the prior issues, then that means proof-of-stake, proof-of-authority, etc., don't actually matter. The governance of the system is the real proof-of-authority. And, of course, the governance of the system can then still run all the aforementioned attacks.

## Where might the properties of sidechains be especially useful?

Aside from the auxiliary properties of sidechains, such as faster block times leading to better UX (though databases give this too ;)), there are some situations where the specific properties of sidechains are arguably especially well suited to the desired properties of the system. For example:

1. If you specifically want

N-M

validators to be able to perform arbitrary state transitions. Enterprise applications who want to have a master control switch are an example.

1. Where M = 0

and you want N

validators to be able to perform arbitrary state transitions. For example, in a 4 party game. Though one issue here is that 1 validator can unilaterally halt the chain.

## Final thoughts

It used to be the case that sidechains were the only viable solution for certain use cases that wanted to retain a level of Ethereum compatibility and interoperability. Now, as other layer 2 scaling solutions mature, it is a good time to consider how sidechains can be made more compatible with those solutions.

Some additional features / properties that would be great for sidechains to incorporate:

1. Implement mass migrations without a fee to ensure users can exit without being "stuck" due to costs.
2. Replace the leader election mechanism with something with stronger anti-censorship properties (proof-of-stake seems to be the wrong

direction to go in here – seeAgainst proof of stake for [zk/op]rollup leader election)

1. Require coordinators to place the diff between two states on chain.
2. Add fraud proofs to prevent illegal state transitions.

As optimistic rollup tech and the optimistic VM (OVM) mature, the tradeoff space for projects will change. Thus, now seems like a good time to refresh on sidechain properties and their associated tradeoffs.