Polkadot.

Slashing in sharded systems

Alistair Stewart, Web 3 Foundation

Polkadot parts are joint work with Jeff Burges
and Handan Kilinc Alper

Proof of stake and slashing

Polkadot.

A permissionless blockchain should be secure if enough participants are rational.

They typically rewards consensus participants for correctly participating.

Proof of stake design goal: If most stake is in the hands of rational and (somewhat) active actors, then the system is secure.

Proof of stake protocols can also punish participants who incorrectly participate in the consensus, by slashing them i.e. taking away their stake.

Rationality and honesty assumptions

Polkadot.

We would like to understand the correctness of complex protocols under rationality alone

But typically we cannot do this. So instead we make honesty assumptions, like that >2/3 of participants/power are honest

Honesty assumptions alone are enough to show the correctness of protocols.

Can we get more by making economic assumptions as well as honesty assumptions?

Is slashing a good idea? I

Polkadot.

Suppose there is a BFT protocol to secure a blockchain, which is secure if >1/2 (synchronously safe case) or >2/3 (asynchronously safe case) of consensus participants/power are honest

If >1/2 or >2/3 of participants/power are honest then we only suffer a small penalty to performance from dishonest participants

If >1/2 or >2/3 of participants/power are dishonest and colluding, then they can make any valid chain canonical.

They can censor the chain. In particular they won't include any reports that cause them to be slashed.

In the asynchronously safe case, If >1/3 and <2/3 of participants/power are dishonest and colluding, then they can stop the protocol from deciding anything.

Is slashing a good idea? I

Polkadot.

- A continuously successful attack does not result in anyone being slashed.
- Slashing incorrect behaviour disincentives it when there is no successful attack
- Slashing disincentives participation in staking.

- “Professorcoins” do not slash e.g. Algorand, Avalanche, Cardano

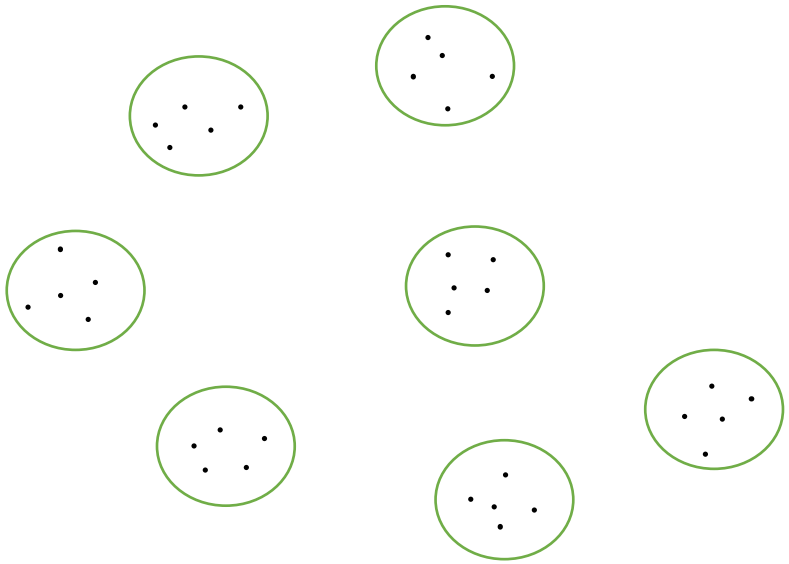
Slashing in sharded or interoperating protocols

Polkadot.

- Misbehaviour on one chain can be reported and slashed on another chain.
- Layer 2 protocols – most transactions off main chain, only use main chain for disputes
- Sharding with a main (relay, beacon, hub) chain
- Bridges – can have relay on one chain to follow consensus on the other and consensus misbehaviour on the chain with the relay can be reported on the original chain.

Sharding with only honesty assumptions

A system has s shards, n participants with equal power called validators, m=n/s participants per shard.



Every time period, each shard has m participants chosen at random with all subsets of m participants having equal probability. Then each shard uses these in a BFT protocol. We consider the system for t such time periods.

Sharding with only honesty assumptions: Analysis

Each shard needs > ½ or >2/3 honest validators to function correctly. We assume that the whole system has >2/3 or >5/6 honest validators respectively.

Model as m validators are bad independently with probability ½ or 2/3. Then we can get a bound of the probability that a particular shard does not function correctly in a particular time period using Hoeffding's inequality as at most:

$$\exp(-2m\,(1/6)^2)=\exp(m/18)$$

For s shards to be correct in each of t time periods except with probability p we need

$$m > 18 \ln (st/p)$$

e.g. for 100 shards, 365x40 time periods, p=0.01, we'd want

$$m > 338.4$$

Sharding with slashing

Polkadot.

As with layer 2 systems we can use a central (unscalable) chain to deal with disputes.

If just one validator on a shard is honest, they can report misbehaviour to the central chain. Either they include a proof or a complaint causes everyone to check.

This breaks scalability, but this is paid for by slashing someone. We do not believe what shard validators decide for some challenge period.

For s shards to be correct in each of t time periods except with probability p we need

$2/3^m < p/st$

e.g. for 100 shards, 365x40 time periods, p=0.01, we'd want

$m > 46.3$

Sharding with slashing: issues

Polkadot.

Availability – if the one validator does not have the data in the challenge period, they cannot point out its invalidity.

> We can make availability more robust using erasure coding. The protocol distributes pieces of each shard to every validator, and they agree it is available before the challenge period starts.

Attacker knows when to mount an attack: Just wait until all m validators on a shard are selected from a malicious colluding set.

Cost of an attack: It only costs an attacker an O(1/s) fraction of stake to mount an attack.

Is it even possible to be more secure than than O(1/s) attack cost?

Polkadot.

If

1. Checking validity is atomic
2. Each validator can only check O(1) blocks

Then an average of O(1/s) validators check each shard block. If all validators on the least checked shard collude then they can attack the system but only be slashed with total cost O(S/s) where S is the total stake in the system.

Is slashing a good idea? II

Polkadot.

Rationality is Self-Defeating in Permissionless Systems – Bryan Ford and Rainer Böhme

The total amount of money in the world, M >> Amount of stake in the system, S >> maximum total reward/punishment for actor(s) capable of attacking the system, V

Therefore it may at some point be worth someone paying V to take down the system or steal >V money.

The expected cost of an attack

Polkadot.

$$\text{Expected cost} = \text{Cost of an attack attempt} \times \frac{\text{detection probability}}{\text{success probability}}$$

Make detection probability high ≈ 1. Then success probability needs only to be $\Omega(1/s)$. Elimination the ln t term helps a lot.

Using unknown participants

Polkadot.

An adversary attacking participants can only do so if they know who they are.

Can choose participants via proof of work or verifiable random functions (e.g. Algorand, Ouroboros family). Identity is unkown until they participate.

Then we can get a protocol that works under honesty assumptions even with a few participants deciding.

But if every validator runs hacked software, then they can rationally take bribes offered by a smart contract on another chain.

Availability and validity in Polkadot

Polkadot.

1. m backing validators assigned to the shard claim a block is correct

2. Erasure coded pieces are distributed to all validators. 2/3 agree that this happened correctly

3. An average of m’ approval validators choose to check this shard if their VRF tells them to.

4. The backing validators acquire the erasure coded pieces for the block, reconstruct it, check its validity and broadcast the result

5. If any say it is invalid, escalate, otherwise after some time, agree on it

Availability and validity in Polkadot: Analysis

Out of the 2n/3 honest validators, the number assigned to this shard is distributed as

$$\text{Binomial}(2n/3, m'/n) \sim \text{Poisson}(2m'/3)$$

Therefore the probability that none check is approximately

$$\exp(-2m'/3)$$

The expected cost of an attack is

$$(m\ S/n)\ \exp(2m'/3)$$

If $m=m'=n/2s$, then the cost is more than S when

$$m' > (3/2)\ln(2s)$$

So when s=100, we need

$$m + m' > 13.8$$

Can a similar design work with bridges?

Polkadot.

Imagine trying to follow a BFT consensus protocol on another chain. Checking signatures or even including them and validator's public keys, can be expensive so we want to reduce the number we include and check.

1. A transaction claims that >2n/3 validators signed a statement s. It includes a bitfield of claimed signers and one explicit signature.

2. A smart contract checks the signature and stores the claim

3. A second transaction includes signatures of m random validators, selected by some on-chain randomness. The claim is accepted if all signed it

If >n/3 validators sign it, then this will accept with probability 2^{-m}.
A larger m allows us to deal with biasable randomness, e.g. PoW block hashes under assumptions about consensus participants on the checking chain.

Conclusion

Polkadot.

- It makes sense to combine economic and honesty assumptions.
- This allows more scaling
- Consensus systems not existing in isolation makes rationality assumptions more interesting!