Contract Enforcement and Decentralized Consensus: The Case of Slashing*

Zhiguo He† Jiasun Li‡

October 1, 2022

Abstract

Many new blockchain applications rely on the "stake-and-slash" mechanism to align incentives. We point out that the design of such "contracting" problems cannot be detached from the details of the decentralized consensus formation process. To illustrate our theoretical argument, we empirically investigate Ethereum's beacon chain – an upgrade to a proof-of-stake system. Based on data from Beaconcha.in, the leading Ethereum beacon chain explorer, we find that more than 75% of Byzantine actions have dodged penalty (slashing). Ongoing research is further investigating whether the finding reflects flaws in Ethereum's incentive design or bugs from a major blockchain explorer.

Keywords: Blockchain, Byzantine fault tolerance, Contract theory, Distributed consensus

*Preliminary and do not cite. We thank Ian Gao for excellent research assistance as well as seminar participants at Tsinghua University and Luohan Academy for helpful comments. We are grateful for research grants from the Paris-Dauphine Partnership Foundation. Zhiguo He acknowledges financial support from the John E. Jeuck Endowment at the University of Chicago Booth School of Business.

†zhiguo.he@chicagobooth.edu. Booth School of Business, University of Chicago and NBER, 5807 South Woodlawn Ave, Chicago, IL 60637.

‡jli29@gmu.edu. George Mason University, 4400 University Drive, MSN 1B8, Fairfax, VA 22030, USA.

1 Introduction

The success of proof-of-work (PoW) based Nakamoto consensus adopted by Bitcoin, together with its shortcomings in processing throughput and energy consumption has in recent years motivated research into alternative consensus protocols. Particular interest has been around applying traditional Byzantine Fault Tolerance protocols to permissionless blockchains, with added economic incentives based on proof-of-stake (PoS). Unlike in proof-of-work (PoW) blockchains where miners devote computing powers for the right to propose new blocks (which indirectly vote for previous blocks), in a PoS blockchain, validators stake cryptoassets for the right to propose and vote for new blocks. PoS blockchains incentivize participation by rewarding validators for desirable behaviors just as PoW blockchains do to miner, and deter misbehavior by "slashing" misbehaving validators' "stakes."[1] Therefore, a validator's "stake" serves to both determine her probability of being selected to propose new blocks and discipline her as a deposit of "collaterals."

Due to the decentralized nature of a PoS blockchain, its reward/penalty design brings new questions to the field of contract theory. Most existing contract theory models build on the assumption that there exists a trusted third-party (e.g. the court) who can truthfully enforce contracts. For example, the introduction (Page 3) of Bolton and Dewatripont (2005) explicitly states that

> "*The benchmark contracting situation that we will consider in this book is one ... with a well-functioning legal system. Under such a system, any contract the parties decide to write will be enforced perfectly by a court ... We shall assume throughout most of the book that the contracting parties do not need to worry about whether the courts are able or willing to enforce the terms of the contract precisely.*"

However, in blockchain applications no centralized trusted third-party exists, and reward/penalty protocols have to be executed based on decentralized consensus, whose reliability in turn relies on whether reward/penalty protocols can be perfectly executed. Such an interdependence loop suggests that one has to explicitly consider the consensus formation process in analyzing the soundness of a PoS blockchain's reward/penalty protocol.

[1] We use the word "slashing" in a liberal sense here to describe any explicit on-chain penalties; Ethereum 2.0 reserves slashing to only a subset of egregious misbehavior, as will be detailed in Section 2.

In this paper, we take a first look at this issue by analyzing the consensus formation processes in Ethereum 2.0's Beacon chain, which is part of the ongoing upgrade in Ethereum from PoW to PoS. After explaining how the new system works in Section 2 and providing a theoretical framework in Section 3, we document two empirical facts in Section 4. First, we document plenty of slashing events on Ethereum 2.0's Beacon chain, suggesting the presence of off-equilibrium deviations that are supposedly not to exist in standard optimal contracting models. Second, we detect a handful of "slashable" misbehavior that are left unslashed (by the time this paper is written – we expect validators to slash those uncaught violations once they become aware of results), even though theoretically a well-enforced reward/penalty contract is expected to catch all slashable misbehavior.

Our results point to a limitation of "smart contracts" regarding the size of contractible variables.[2] Contrary to what many economists' perception that smart contracts effectively automates reward/penalties based on arbitrary input variables, we highlight that the contractible variables for smart contract in practice may be restricted to only ones that can reach consensus (i.e. on-chain values). Therefore, our paper points out the nuanced interaction between contract/protocol design and the formation of distributed consensus.

While our analysis mainly focuses on Ethereum 2.0, the lesson goes more generally to other blockchain applications (or decentralized applications in general). Not only does many other proof-of-stake blockchains feature the "stake-and-slash" mechanism built on ex post whistleblowing, similar techniques have also been adopted in Layer 2 solutions such as optimistic rollups or (earlier) Plasma. Our results calls for more explicit incentive analysis into these new design ideas.

Literature Our papers contributes to multiple strands of the literature.

First, we add to the emerging literature on incentive analysis of BFT-based consensus protocols. Halaburda, He and Li (2021) recognize that non-Byzantine nodes do not need to follow the protocol if they do not find it beneficial, and develop an economic framework based on ambiguity aversion to analyze non-Byzantine nodes' incentives in the consensus formation process. Amoussou-Guenou

[2]By smart contract we refer to a liberal interpretation as deterministic computer codes that run on a decentralized blockchain, according to the definition put forward by IBM: "*Smart contracts are simply programs stored on a blockchain that run when predetermined conditions are met. They typically are used to automate the execution of an agreement so that all participants can be immediately certain of the outcome, without any intermediary's involvement or time loss.*"

et al. (2020) also conduct an incentive analysis in BFT protocols by assuming that it is costly for nodes to check the validity of the proposed message and send the confirmation to other nodes. Auer, Monnet and Shin (2021) also assume costly communication and study consensus exclusively among rational nodes who may potentially bribed, and derive conditions when the nodes would find it more beneficial to follow the protocol than to take the bribe. Benhaim, Hemenway Falk and Tsoukalas (2021) analyze incentives in the committee formation process in the context of delegated proof-of-stake mechanism.

Our paper also relates to studies on the incentives within proof-of-work (PoW) protocols (e.g., Budish (2018), Biais et al. (2019), Leshno and Strack (2020), Pagnotta (2020), Hinzen, John and Saleh (2020), Cong, He and Li (2021)), and similarly in other permissionless consensus protocols such as proof of stake (e.g, Saleh (2021) and John, Rivera and Saleh (2021)).[3] Since participating in PoW blockchains incurs hardware and electricity costs, PoW blockchains penalize misbehaving miners by forfeited "work". The lack of "costly work" in PoS blockchains is commonly known as the "nothing-at-stake" problem. Under specific assumptions, Saleh (2021) shows that the "nothing-at-stake" problem may be mitigated when validators recognize the negative impact on cryptoasset values from misbehaving, while the "stake-and-slash" mechanism has been adopted in practice to encounter the "nothing-at-stake" problem as well as other misbehavior.

The economic incentives in Byzantine fault tolerant consensus protocols builds on a large computer science literature which starts with Lamport, Shostak and Pease (1982), who formulated the Byzantine generals problem and showed that consensus is possible. Castro and Liskov (1999) further streamline the consensus algorithm as a practical Byzantine fault tolerant (PBFT) mechanism. More recent developments in BFT protocols include Buterin and Griffith (2017), Buchman (2016), Pass and Shi (2018), Yin et al. (2018), etc. See Shi (2020) for a summary.

Lastly, our empirical results also relates the forensics literature in economics (e.g. Dyck, Morse and Zingales (2021)). Specific to forensics in the crypto context, Griffin and Shams (2020) relate the 2017 bitcoin bubble to Tether issuance from a single large bitcoin address; Gandal et al. (2017) relate the 2013 Bitcoin bubble to price manipulation on the now defunct Mt.Gox Bitcoin exchange,

[3]For other papers that study the broader implications of blockchain technology, see Cong and He (2019), Li and Mann (2018), and Abadi and Brunnermeier (2018), among others.

while Aloosh and Li (2021) point to direct evidence of volume-inflating wash trading Mt.Gox. Cong et al. (2020) and Amiram, Lyandres and Rabetti (2020) develop techniques to statistically infer wash trading, while Li, Shin and Wang (2019) provide direct evidence of pump-and-dump schemes in the cryptocurrency market using communication records on Telegram.

2 Ethereum 2.0: A Brief Overview of Technology Background

For concreteness, we cast our theoretical discussion and empirical analysis in the specific example of Ethereum 2.0, which is an ongoing upgrade to the current Ethereum blockchain (Ethereum 1.0). The upgrade aims to convert the existing PoW mechanism to PoS and further scale up Ethereum's transaction throughput (via "sharding"). The upgrade is planned to take place in multiple phases, with phase 0 creating a new PoS based blockchain known as the Beacon chain. The Beacon chain went online on Dec 1, 2020 and has been operating ever since. It is planned for additional changes to be brought to the Beacon chain and the current Ethereum 1.0 chain to be merged to the Beacon chain in subsequent phases. While Ethereum 2.0 is still an ongoing project, its PoS mechanism of will follow the Beacon chain which has been in operation for more than a year.

Figure 1 gives an overview of the Beacon chain operation. At a high level, the PoS mechanism in Beacon chain works as follows:

As a permissionless blockchain, anyone can stake 32 ETH and become an Ethereum 2.0 validator to participate in the Beacon chain's consensus formation process, which proceeds in time units known as *epochs*. Before an epoch starts, the set of active validators are determined and pseudo-randomly assigned to their respective roles: Some validators are chosen to propose new blocks while all validators (including the proposers) are assigned to make attestations (votes). The assignment makes sure that within a given epoch, each validator has the right to attest once and only once, and each chosen proposer has the additional right to propose one and only one new block.[4]

Every proposed block contains certain history of the Beacon chain. For example, it may include past attestations or occasionally slashing violation evidences (to be detailed later). Like blocks in

[4]Specifically, an epoch is divided into 32 *slots* each lasting for 12 seconds. In a given epoch, 32 out of all active validators are chosen as proposers, with each slot having one proposer. All validators (including the proposers) are subdivided into 32 groups so that each group of validators are assigned to attest one of the 32 slots.

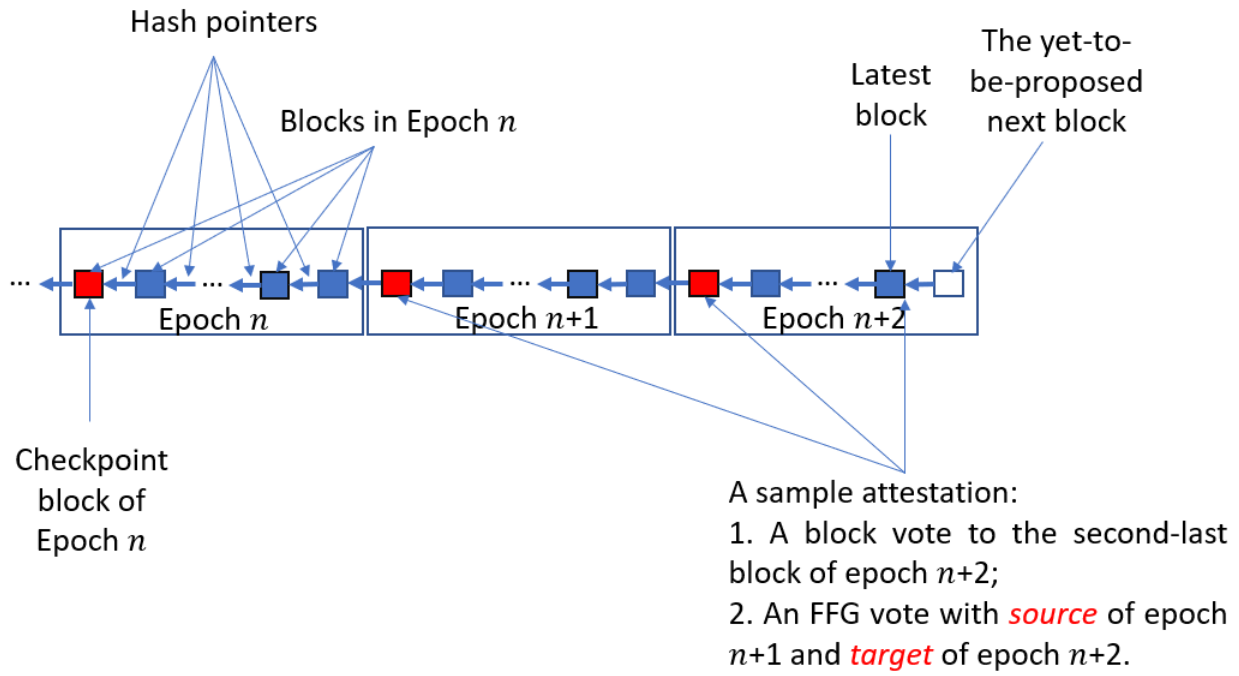

Figure 1: An Illustration of the Beacon Chain Structure

This figure illustrates the structure of the Beacon chain in an ideal environment. Each epoch contains a sequence of blocks proposed by pre-scheduled block proposers. Blocks are connect with each other via hash pointers. The first block of each epoch is denoted as the checkpoint of the epoch. Within a given epoch, each validator makes one and only one attestation. In the figure, a validator makes an attestation that votes for the second-last block of epoch $n+3$ and FFG votes for source $n+2$ and target $n+3$.

Bitcoin (as well as all other blockchain systems), each block also contains the hash of a previous block. Although currently Beacon chain blocks do not include additional transactions, they may further include references to other transactions, once the Ethereum 2.0 update is complete. Finally, as shown in Figure 1, each epoch also defines a *checkpoint* block, which is typically the first block in the epoch.[5]

An attestation contains two types of votes. First, it indicates which newly proposed block it votes for.[6] Attesting to a block indicates an endorsement of the block as the latest block. Second,

[5]Or more precisely, the block proposed in the first slot of the epoch. If the block in the first slot is missing, then the checkpoint is defined as the latest preceding block (which may belong to a previous epoch).

[6]Ideally, all validators assigned to attest in a particular slot vote for the block proposed in the same slot. However, due to network latency, some validators may have not received the current-slot block before the slot expires, and these validators may instead vote for blocks proposed in earlier slots.

in addition to voting for a new block proposal, each attestation also additionally includes an *FFG vote* for checkpoints.[7] An FFG vote specifies both a *target* checkpoint and a *source* checkpoint, with the latter necessarily proceeding the former. While FFG votes do not have apparent analogies in economics or to Nakamoto consensus (as adopted by Bitcoin and Ethereum 1.0), they can be understood at a high level of as a specific type of multi-round voting messages in the spirit of Byzantine Fault Tolerance (BFT) protocols to help finalize blocks. [8] We will explain BFT protocols in the next section.

All proposals and attestations are messages broadcast to peers.[9] Compliant proposals and attestations will bring rewards of newly minted Ether once the epoch ends.

2.1 Slashing conditions

To ensure the security of the blockchain, all validators are expected to comply with certain rules when proposing new blocks or making attestations. Roughly speaking, these rules require validators to never contradict with themselves. Violators may potentially be caught and slashed, that is, they will be deprived of the privilege to act as a validator (and thus collect rewards) anymore, and their stakes will be deducted according to a predefined rule. These violations are "double proposal", "double vote", and "surround vote".

We now explain in detail the conditions for each slashable violation.

Double proposal. A "double proposal" violation happens when a proposer proposes two conflicting blocks. A double proposal resembles the proposer "equivocating" different messages in a BFT context, which is prevented (with an overwhelming probability) in Bitcoin by the "proof-of-work" requirement that makes it costly to create another proposal. Figure 2 illustrates an example of double proposals;

[7] FFG stands for Casper: the Friendly Finality Gadget, which is a protocol designed on top of a running blockchain for finalizing blocks in a Byzantine Fault Tolerance (BFT) fashion. See Buterin and Griffith (2017).

[8] Specifically, once a checkpoint has gathered FFG votes from more than $\frac{2}{3}$ of all validators (reaching a supermajority), the checkpoint becomes justified. Once the immediately succeeding checkpoint of a previously justified checkpoint becomes justified, the previously justified checkpoint becomes finalized.

[9] In practice, to reduce bandwidth/storage usage, validators are further grouped into several committees so that many communications only happen within committees. Ethereum 2.0 adopts the BLS threshold signature (Boneh, Lynn and Shacham (2004)) so that within-committee communications are aggregated for cross-committee communications. That said, the simplifying statement above is sufficient for our further discussions so we spare the details.

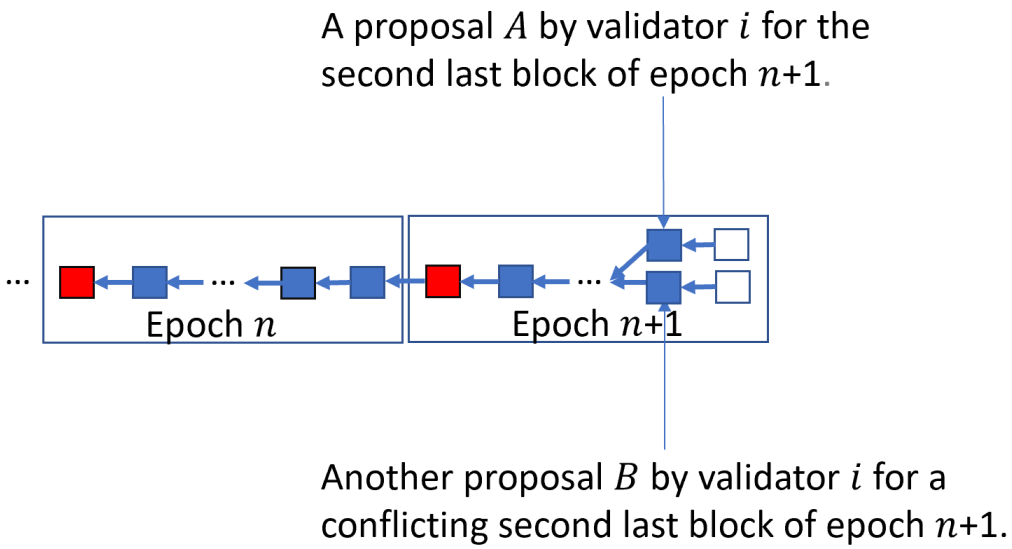

Figure 2: Double proposals illustrations

This figure illustrates double proposals: Validator i makes two proposals A and B with two conflicting blocks at the time. If we see a validator created such pair of proposals, this validator needs to be slashed.

Double votes. A "double vote" violation happens when a validator ever votes for two different blocks in her turn within the same epoch. Double votes for block proposals resembles appending conflicting existing blocks in Bitcoin, which is also prevented (with an overwhelming probability) by the "proof-of-work" requirement since in Bitcoin voting for a previous block happens concurrently with making a new block proposal. Surround votes or double FFG votes resemble "dishonest" voting behaviors in BFT protocols and do not have an obvious analogy within the Bitcoin context. Figure 3 illustrates an example of double votes;[10]

Surround votes A "surround vote" violation happens when a validator ever casts two FFG votes A and B so that Source(A)< Source(B)<Target(B)<Target(A), where Source(A) and Target(A) denote the epoch numbers of the source and target in FFG vote A, respectively, and Source(B) and

[10]The definition here follows the practice of observed slashing incidents of attester violations. The community sometimes also use "double votes" to describe the behavior of casting FFG votes for two conflicting target checkpoints at the same epoch (e.g. https://ethos.dev/beacon-chain/). Since such violations necessarily require existing double proposals, we will group them into double proposals. Figure 7 in the appendix will give one illustration of this case.

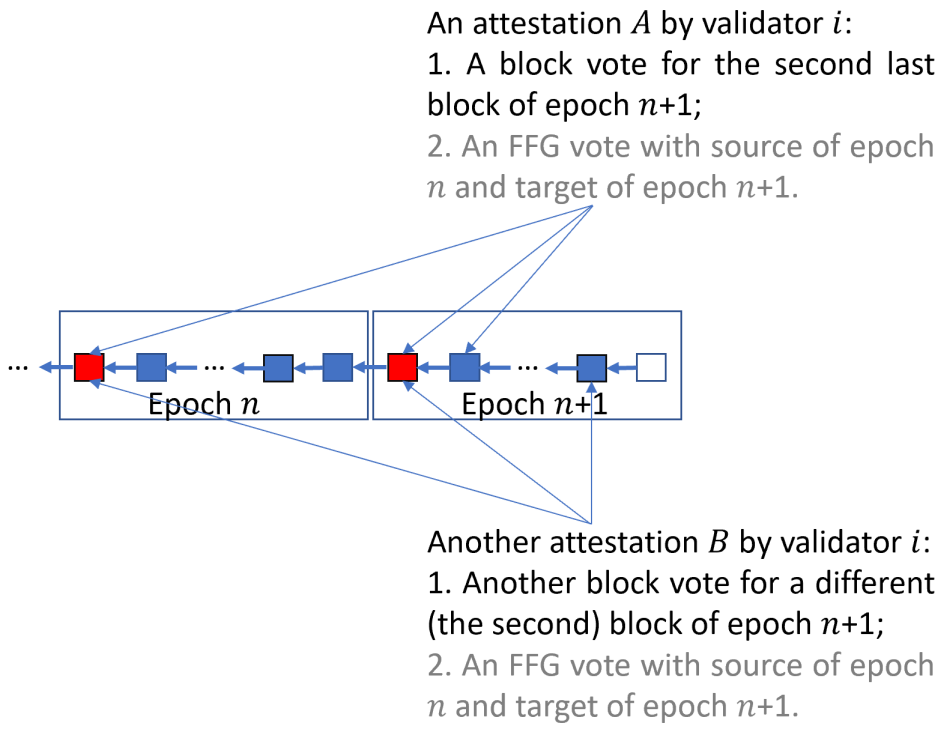

Figure 3: Double votes illustrations

This figure illustrates double votes for new block proposals: attestation A votes for the second block of epoch $n+1$, while attestation B votes for a different (the second last) block of epoch $n+1$. In this case, validator i vote twice with different values within the same epoch, and thus attestations A and B constitute of double votes. If we see a validator created such pair of attestations, this validator needs to be slashed.

Target(B) denote the epoch numbers of the source and target in FFG vote B, respectively (recall the structure of an FFG vote from Figure 1). Figure 4 illustrates an example of surround votes.

A rough intuition for slashing surround or double FFG votes is as follows. In BFT protocols, "honest" behaviors, that is, to not deviate from the protocol's specified forwarding and voting strategies, ensure any record that has reached consensus to never be overturned under certain security conditions, say more than two-thirds of nodes are honest (see e.g. Castro and Liskov (1999); in contrast, Bitcoin does not have such a feature as Bitcoin blocks are never 100% finalized). In the context of Beacon chain, Buterin and Griffith (2017) show that for two conflicting checkpoints to ever get finalized, it necessarily requires more than one-third of validators to have cast two conflicting FFG votes that constitute a pair of either double votes or surround votes. Therefore, if

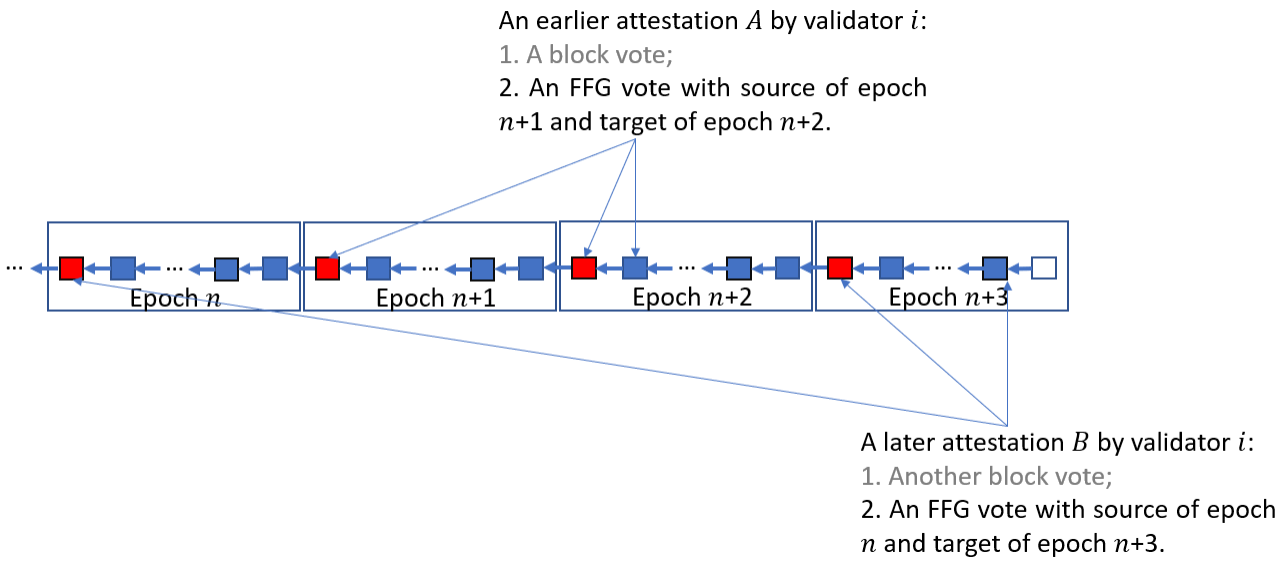

Figure 4: Surround vote illustrations

This figure illustrates an FFG vote "surrounding" a previous FFG vote: validator i's attestation A during epoch $n+2$ specifies a source of epoch $n+1$ and a target of epoch $n+2$, while later during epoch $n+3$ the same validator sends a new attestation B which specifies a source of epoch n and target of epoch $n+3$. Then attestation B surrounds attestation A. If we see a validator create such pair of attestations, this validator needs to be slashed. Alternatively, a new FFG vote may also be "surrounded" by a previous FFG vote. Figure 8 in the Appendix will give further illustrations.

fewer than one-third of validators commit such violations, then the Beacon chain will be "safe" in the sense that no conflicting checkpoints will ever be finalized. The threat of slashing aims to deter any validator from committing these violations, and thus ensure the "$< \frac{1}{3}$" condition. Appendix B will explain in more detail why the conditions ensure chain safety.

2.2 Slashing detection in practice

When any of the above violation is committed, evidence of violation can be gathered. A proposer who gathers such evidence may include them in her proposed block to trigger slashing of the offending validator. Of course, for the slashing to be effective, the proposed block has to reach consensus, too.

It is important to point out that when a validator is found to have committed some slashable offenses, it indisputably indicates that the validator has done something different from recommended

practices. As https://ethos.dev/beacon-chain/ highlights,

> "*A validator is in total control to avoid getting slashed: it only needs to remember what it has signed. An honest validator cannot be slashed by the actions of other validators. As long as a validator does not sign a conflicting attestation or proposal, the validator cannot be slashed.*"

In practice, all available Ethereum 2.0 client software by default implements slashing protection,[11] which maintains a local database of all previous proposals/atttestations made the validator, so that any new proposal/atttestation will be compared with the local database before being broadcast to other validators. By filtering through the local database, the slashing protection mechanism ensures a validator to never send a slashable proposal/atttestation, at the cost of delaying sending new proposal/atttestation (and potentially limiting the validator's performance). Therefore, slashing protection resembles risk control mechanisms in many financial institutions – it safeguards against risk-taking by the institution (analogous to the validator) and prevents externalities to the financial market in general (analogous to the Beacon chain in general), while potentially restricting the institution's profitability. In this regard, the occurrence of a slashable violation indicates that the committing validator has either disabled the default slashing protection or have otherwise customized its client software at the cost of increased slashing probability (and in turn at the cost of the Beacon chain's security).

3 Contract Theory and Connection to Slashing on Blockchain

3.1 Contract theory revisited

From an economist's perspective, a blockchain's reward/penalty protocol effectively specifies a contract, which maps validators' actions into certain pecuniary reward/penalty functions. In the familiar principal-agent setting of Holmström (1979), when an agent takes an action a, a contract specifies that the agent receives $\Pi\left(y\left(a,\epsilon\right)\right)$ where $y\left(a,\epsilon\right)\in Y$ is some noisy signal of action a and ϵ is typically assumed to be exogenously given. For instance, Holmstrom and Milgrom (1987) assume

[11] See e.g. the Prysm Documentation or the Lighthouse Book.

that $y(a, \epsilon) = a + \epsilon$. An implicit assumption in the above specifications, which prevails in the field of "complete contract theory," is that the observable signal $y(a, \epsilon) \in Y$ is contractible.

3.2 Mapping our framework to the Ethereum 2.0 blockchain

Mapping the above classic theory into a (decentralized) blockchain context, the reward/penalty protocol works as follows. When validators take a set of actions a, these actions will induce a set of outcomes $y \in Y$ so that validators get paid or penalized by $\pi(y)$. We highlight that each element in Y has to become consensus (i.e., they are on-chain records) in order to be "contractible." Here, the contractible signal $y(a, \epsilon)$ could involve sophisticated equilibrium interactions; for instance, if the evidence of misbehavior does not become consensus, then it is impossible for misbehaving validators to be punished.

The Ethereum 2.0 protocol explained in Section 2 specifies some desirable behavior a for a compliant validator. Then any deviations from a can be conceptualized as a misbehavior a'. For example, a' may include the action of disabling slashing protection, which increases the probability of incurring slashable offenses. In this example, the occurrence of an actual slashable offense can be interpreted as the noisy signal $y(a', \epsilon)$.

Furthermore, the decentralized nature of blockchain implies that not necessarily any $y(a', \epsilon)$ will trigger penalty or reward. In Ethereum 2.0's current implementation, the evidence of misbehavior a'_i, or the signal of misbehavior $y(a', \epsilon)$, has to be first assembled by some validator (known as the whistleblower) and then included by the proposer of a new block. For instance, as explained by the Rewards and Penalties on Ethereum 2.0:[12]

> *In all these cases, the offender needs to be caught in order for the slashing process to be triggered. The whistleblowing validator will create and propagate a specific message containing the offense, for a proposer to include it in a block.*

For slashing to take place, a block that embeds "wistleblowed" misbehavior has to reach consensus (so the whistleblowing message is included in the blockchain). However, if the block is orphaned

[12] For actual deployment of the idea here, see e.g. Beacon Chain spec.

or simply discarded by all validators except its proposer, then slashing will not take place. Therefore, not all misbehaving validators will necessarily be slashed. In an even more primitive step, for a slashable violation to be actually slashed, it has be first detected by a wistleblower. If such detection does not take place (either due to costly detection or inadequate incentives),[13] violations may be left unintended and not slashed. Our empirical analysis to follow will show that there are indeed uncaught slashable violations.

4 Slashed and Missed Misbehavior

To investigate the well-functioning of the reward/penalty mechanism and the performance of slashing enforcement, we conduct an empirical analysis of the Beacon chain. We collect data from beaconcha.in, which is one of the official blockchain explorer of the Beacon chain. Our data include all proposal/attestation records as well as orphaned blocks for the first 1.75 million blocks (from the genesis block on Dec 1, 2020 to August 1, 2021). In the following discussions, we first summarize all violations that have been slashed, and then present our findings from the data of violations that have not yet been slashed.

4.1 Recorded slashing incidents

As we have explained above, the Beacon chain relies on slashing mechanism to deter misbehavior. Ideally, if the incentive scheme works perfectly, then we should expect to see no slashable violations to occur at all, as contract theory typically predict no off-equilibrium path deviations. However, in practice slashing events do happen from time to time. Indeed, since its launch in Dec 1st, 2020, we have already witnessed many slashing events. Figure 5 chronicles all slashing incidents in our sample. Appendix C includes a detailed list of all *detected* slashing incidents in our sample.

Overall, as has been expected in the Ethereum community, slashing events tend to be rare occurrences: Out of the first 1.75 million blocks, there are just 156 recorded slashing incidents,

[13] In practice, both channels tend to be at work. For discussions on the resource cost in detecting violations, see e.g. the documentation of Prysm, one of the most popular Ethereum 2.0 client software, which states that "*Slasher ... uses significantly more disk space when running on mainnet.*" The same documment also mentions the lack of incentives to whistleblowers: "*Running a slasher is not meant to be profitable.*"

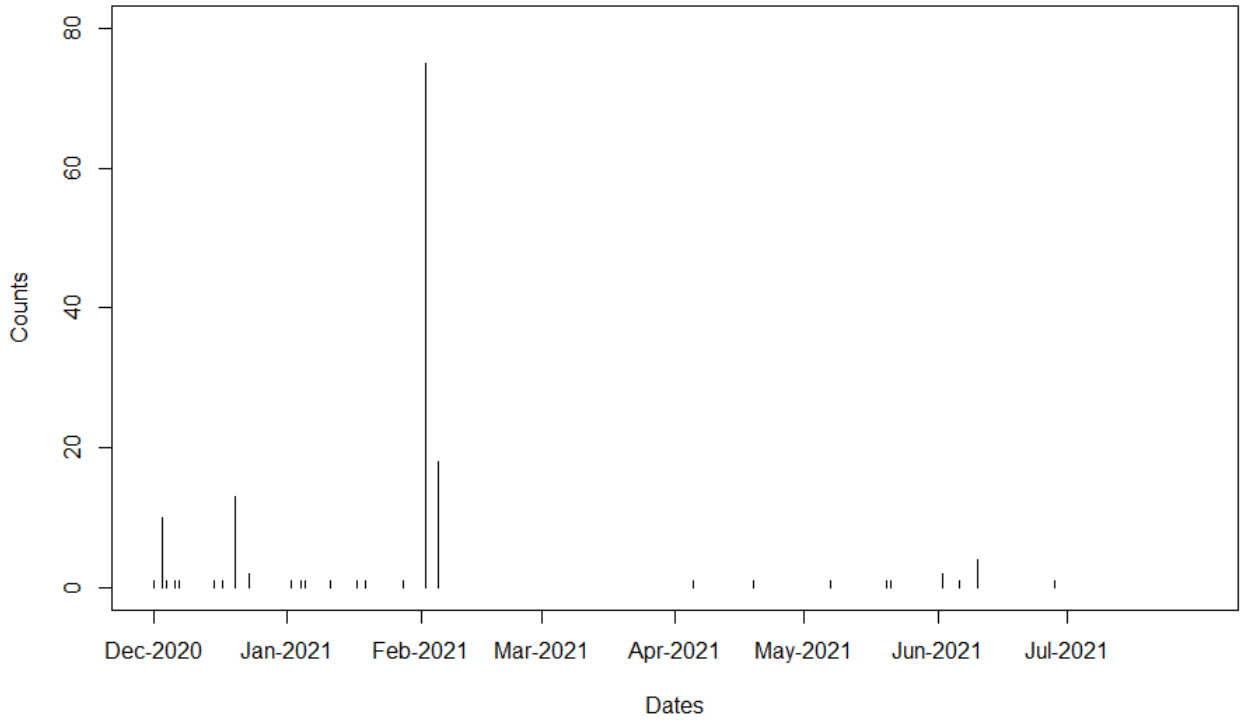

Figure 5: Slashing incidents over time

This figure plots for every day within our sample the count of slashing incidents. The sample includes the first 1.75 million Beacon chain blocks from genesis, which correspond to December 1, 2020 to August 1, 2021.

including 15 proposer violations and 144 attester violations. Furthermore, slashing incidents tend to cluster. For example, out of all slashing incidents, 75 of them happened on the same day. When the Beacon chain is functioning well, there could be prolonged periods during which no slashes take place; yet there could be times when suddenly "many things go wrong." However, it is worth pointing out that the recorded slashing incidents may disguise violations that are not detected and thus (mistakenly) not slashed, as the next section will show.

The slashed violations may indicate intentional attacks to the security of the Beacon chain, or may be due to validators' software misconfiguration like disabling slash protection as mentioned earlier. These observations suggest a limitation of the Beacon chain's incentive design, as a well-designed incentive system should deter rational validators against deviating from prescribed

behaviors, so that no slashing incidents should have been observed.

In the next section, we will show a perhaps even more concerning fact about the Beacon chain's incentive design, in that many more violations were actually left unslashed.

4.2 Unslashed (so far) slashable misbehavior

In the previous section, we document the presence of slashed violations, which indicates that the incentive design of the Beacon chain fails to deter violations completely. In this section, we present a related and perhaps more direct indication of the incentive problem, in that a lot more slashable misbehaving violations actually dodged detection and remain unslashed (until the time this paper is written).

Figure 6 chronicles such unslashed violations in our sample. To facilitate comparison, we plot these unslashed violations along with actual slashing incidents, with the former in dashed red and the latter in solid black. Similar to slashed violations, we also see that unslashed violations tend to cluster. It also seems that unslashed violations tend to increase over time. One possible reason is that as the Beacon chain becomes larger, it becomes more computationally expensive to monitor new proposals/attestations. Appendix D includes a detailed list of all such *unslashed* violations.

The number of unslashed violations is large compared to actual slashes. During our sample, there are 478 unslashed violations, including 404 double votes and 74 surround votes. In comparison, recall that the actual number of slashed attestation violations is 144. Therefore, the numbers indicate that $\frac{478}{478+144}$, or more than 75% attestation violations have dodged detection and thus penalization. This number seriously questions the effectiveness of the "stake-and-slash" mechanism in not only deterring but also detecting misbehaving violations.

Another interesting observation is that the sample of all unslashed violations are exclusively about attestation violations, including both surround votes and double votes, while all proposer violations have been detected and slashed. Likely reasons why only attestations but no proposals dodged detection and slashing include the following: First, proposals in the Beacon chain are fewer in numbers, whereas attestations are come in much greater numbers. As a result, it is relatively easier for validators to monitor the few proposals than a lot more attestations. Second,

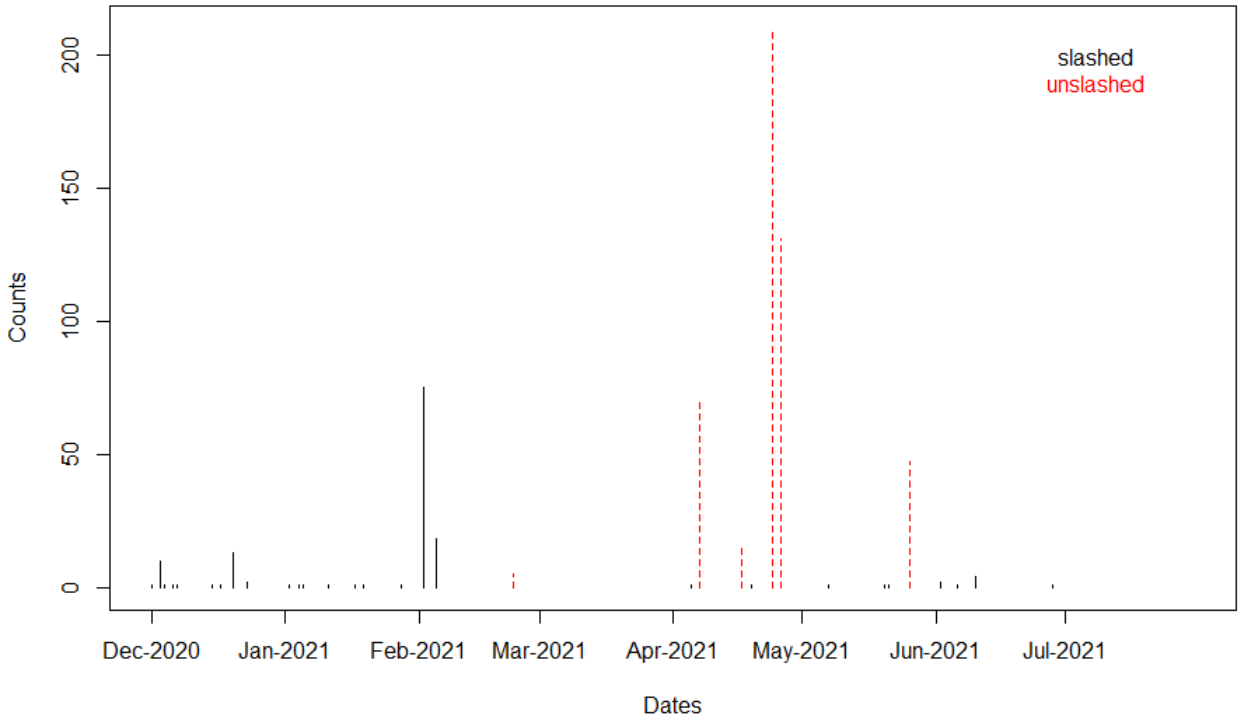

Figure 6: Slashing incidents over time

This figure plots for every day within our sample the count of slashable (yet unslashed) misbehaving violations. We plot these incidents with red dashed bars. For comparison, we also plot slashing incidents in black solid bars. The sample includes the first 1.75 million Beacon chain blocks from genesis, which correspond to December 1, 2020 to August 1, 2021.

and perhaps more importantly, proposals in the Beacon chain are with "real content" whereas attestations are necessary "acknowledgment" steps in the consensus forming process. Therefore, as part of the consensus formation process, all validators constantly listen to new proposals, and it is thus unlikely for a double proposal to dodge the entire validator community's attention; on the other hand, validators do not need to actively check the content of other validators' attestations for them to proceed with validation, presumably making it more likely for an attestation violation to dodge attention.

5 Conclusion

To the extent that a blockchain facilitates "code as the law," the focus of our paper is not about a legislative question, as we take the blockchain protocol (codes) as exogenously given (and common knowledge). Rather, we are interested in a judicial question of enforcing laws in a decentralized consensus process as no (centralized) trusted court exists. The decentralization requirement tends to restrict the set of contractible variables, eroding the enforcement power of codes.

While we cast our discussion in the specific implementation of Ethereum 2.0 for concreteness, the issues discussed in this paper apply more generally. For example, there have been discussions about implementing slashing via smart contracts (in a narrow sense as currently implemented on Ethereum 1.0). However, in practice such smart contracts also need to be triggered to initiate state transitions (that is, evidence of a validator's misbehavior must be fed by some other validtor to the smart contract via a transaction, much like how the whistle-blower sends a message with misbehavior evidences), and the transaction triggering the smart contract must also be included by the block proposer (much like how the block proposer in Ethereum 2.0 needs to include the slashing message in her block). Therefore, implementing slashing via smart contracts would induce the same set of issues discussed in this paper.

References

Abadi, Joseph, and Markus Brunnermeier. 2018. "Blockchain economics." National Bureau of Economic Research. 4

Aloosh, Arash, and Jiasun Li. 2021. "Direct Evidence of Bitcoin Wash Trading." 5

Amiram, Dan, Evgeny Lyandres, and Daniel Rabetti. 2020. "Competition and Product Quality: Fake Trading on Crypto Exchanges." *Available at SSRN 3745617*. 5

Amoussou-Guenou, Yackolley, Bruno Biais, Maria Potop-Butucaru, and Sara Tucci-Piergiovanni. 2020. "Committee-based Blockchains as Games Between Opportunistic players and Adversaries." 3

Auer, Raphael, Cyril Monnet, and Hyun Song Shin. 2021. "Permissioned distributed ledgers and the governance of money." 4

Benhaim, Alon, Brett Hemenway Falk, and Gerry Tsoukalas. 2021. "Scaling Blockchains: Can Elected Committees Help?" *Available at SSRN 3914471*. 4

Biais, Bruno, Christophe Bisiere, Matthieu Bouvard, and Catherine Casamatta. 2019. "The blockchain folk theorem." *Review of Financial Studies*, 32(5): 1662–1715. 4

Boneh, Dan, Ben Lynn, and Hovav Shacham. 2004. "Short signatures from the Weil pairing." *Journal of cryptology*, 17(4): 297–319. 7

Buchman, Ethan. 2016. "Tendermint: Byzantine fault tolerance in the age of blockchains." PhD diss. 4

Budish, Eric. 2018. "The Economic Limits of Bitcoin and the Blockchain." National Bureau of Economic Research. 4

Buterin, Vitalik, and Virgil Griffith. 2017. "Casper the friendly finality gadget." *arXiv preprint arXiv:1710.09437*. 4, 7, 9

Castro, Miguel, and Barbara Liskov. 1999. "Practical Byzantine fault tolerance." *Proceedings of the third symposium on Operating systems design and implementation*, 173–186. 4, 9

Cong, Lin William, and Zhiguo He. 2019. "Blockchain disruption and smart contracts." *The Review of Financial Studies*, 32(5): 1754–1797. 4

Cong, Lin William, Xi Li, Ke Tang, and Yang Yang. 2020. "Crypto Wash Trading." *working paper*. 5

Cong, Lin William, Zhiguo He, and Jiasun Li. 2021. "Decentralized mining in centralized pools." *The Review of Financial Studies*, 34(3): 1191–1235. 4

Dyck, IJ, Adair Morse, and Luigi Zingales. 2021. "How pervasive is corporate fraud?" *Rotman School of Management Working Paper*, , (2222608). 4

Gandal, Neil, JT Hamrick, Tyler Moore, and Tali Oberman. 2017. "Price Manipulation in the Bitcoin Ecosystem." 4

Griffin, John M, and Amin Shams. 2020. "Is Bitcoin really untethered?" *The Journal of Finance*, 75(4): 1913–1964. 4

Halaburda, Hanna, Zhiguo He, and Jiasun Li. 2021. "An Economic Model of Consensus on Distributed Ledgers." National Bureau of Economic Research. 3

Hinzen, Franz J, Kose John, and Fahad Saleh. 2020. "Bitcoin's Fatal Flaw: The Limited Adoption Problem." *NYU Stern School of Business*. 4

Holmström, Bengt. 1979. "Moral hazard and observability." *The Bell journal of economics*, 74–91. 11

Holmstrom, Bengt, and Paul Milgrom. 1987. "Aggregation and linearity in the provision of intertemporal incentives." *Econometrica: Journal of the Econometric Society*, 303–328. 11

John, Kose, Thomas J Rivera, and Fahad Saleh. 2021. "Equilibrium staking levels in a proof-of-stake blockchain." *Available at SSRN 3965599*. 4

Lamport, Leslie, Robert Shostak, and Marshall Pease. 1982. "The Byzantine Generals Problem." *ACM Transactions on Programming Languages and Systems*, 4(3): 382–401. 4

Leshno, Jacob, and Philipp Strack. 2020. "Bitcoin: An impossibility theorem for proof-of-work based protocols." *American Economic Review: Insights*. 4

Li, Jiasun, and William Mann. 2018. "Digital tokens and platform building." 4

Li, Tao, Donghwa Shin, and Baolian Wang. 2019. "Cryptocurrency pump-and-dump schemes." *Available at SSRN 3267041*. 5

Pagnotta, Emiliano. 2020. "Decentralizing money: Bitcoin prices and blockchain security." *Review of Financial Studies (forthcoming)*. 4

Pass, Rafael, and Elaine Shi. 2018. "Thunderella: Blockchains with optimistic instant confirmation." 3–33, Springer. 4

Saleh, Fahad. 2021. "Blockchain without waste: Proof-of-stake." *The Review of financial studies*, 34(3): 1156–1190. 4

Shi, Elaine. 2020. *Foundations of Distributed Consensus and Blockchains.* Book manuscript, Available at https://www.distributedconsensus.net. 4

Yin, Maofan, Dahlia Malkhi, Michael K Reiter, Guy Golan Gueta, and Ittai Abraham. 2018. "HotStuff: BFT consensus in the lens of blockchain." *arXiv preprint arXiv:1803.05069.* 4

Appendix

A Additional double/surround vote illustrations

We give a few additional illustrations of double votes and surround votes in Figure 7 and 8.

B Safety of the FFG protocol

This section provides an intuitive explanation of why the absence of surround or double votes is sufficient for the safety of finalized checkpoints. Indeed, we show that if two conflicting checkpoints ever both get finalized, then more than $\frac{1}{3}$ validators must have cast either surround or double votes.

First, recall from Footnote 8, a checkpoint becomes finalized when its immediate next checkpoint becomes justified, that is, having received more than $\frac{2}{3}$ of FFG votes from all validators as a target. Also recall from Footnote 4 that each epoch is divided into 32 slots. Then Figure 9 illustrates the justification and finalization of checkpoints.

We now prove the argument by contradiction. Suppose two conflicting checkpoints A and B both get finalized (and use $e(A)$ and $e(B)$ to denote the epoch number of checkpoints A and B). Discuss two scenarios: (1) If A and B are for the same epoch, that is, $e(A) = e(B)$, then more than $\frac{2}{3}$ validators have included A as target in their FFG votes, and (not necessarily the same set of)

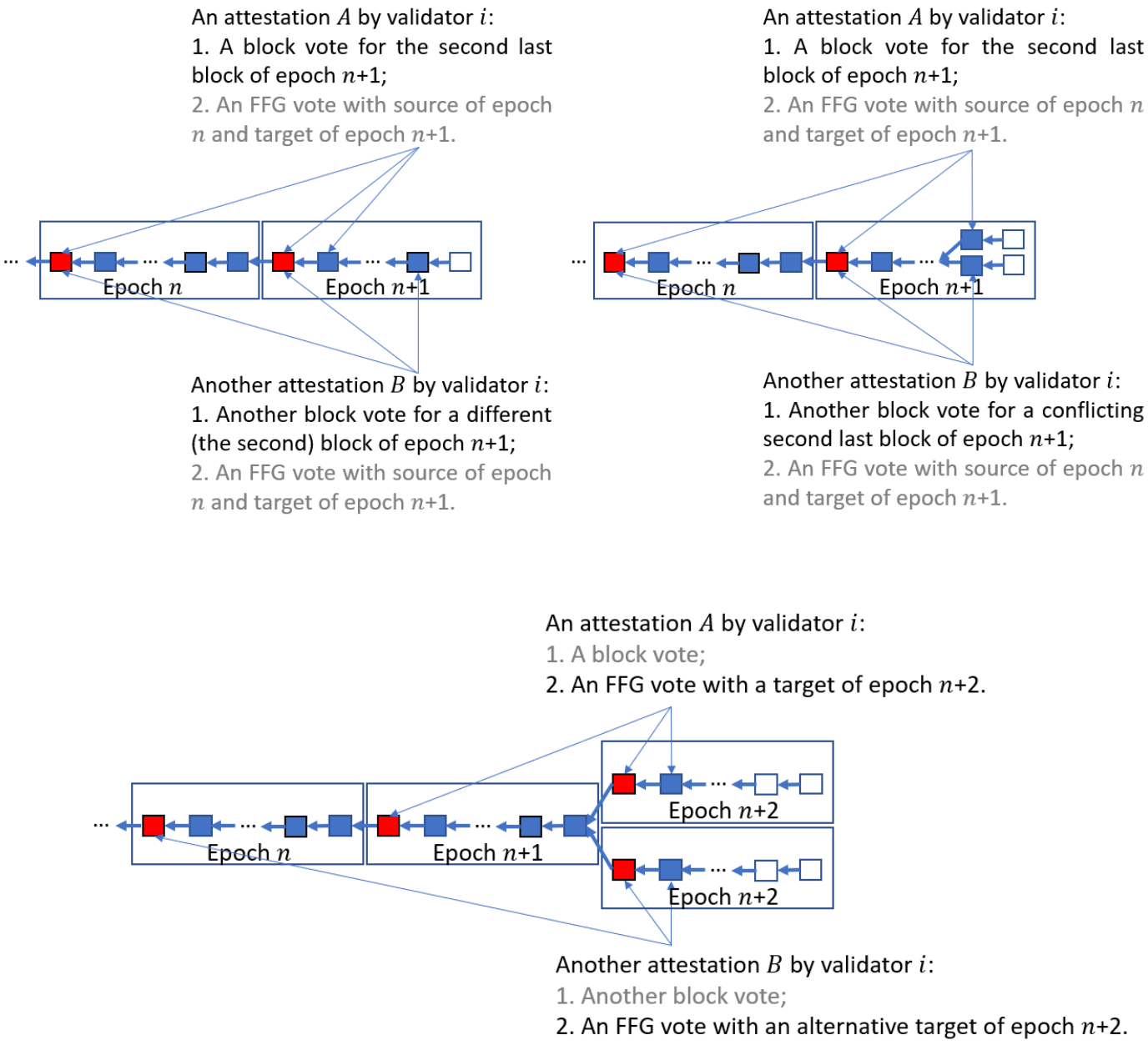

Figure 7: Further illustrations for double vote

The upper figures illustrates double votes for new block proposals: in the upper left figure (which has appeared in Figure 3), attestation A votes for the second block of epoch $n+1$, while attestation B votes for a different (the second last) block of epoch $n+1$; In the upper right figure, attestation A votes for the second last block of epoch $n+1$, while attestation B votes for a conflicting second last block of epoch $n+1$ (a forking block). In both cases, validator i vote twice with different values within the same epoch. The lower figure illustrates a double FFG vote: attestation A specifies a target checkpoint of epoch $n+2$, while attestation B specifies a conflicting target checkpoint also for epoch. In either case, attestations A and B constitute of double votes. If we see a validator created such pair of attestations, this validator needs to be slashed. Note the second and third case in this figure both involve double proposals.

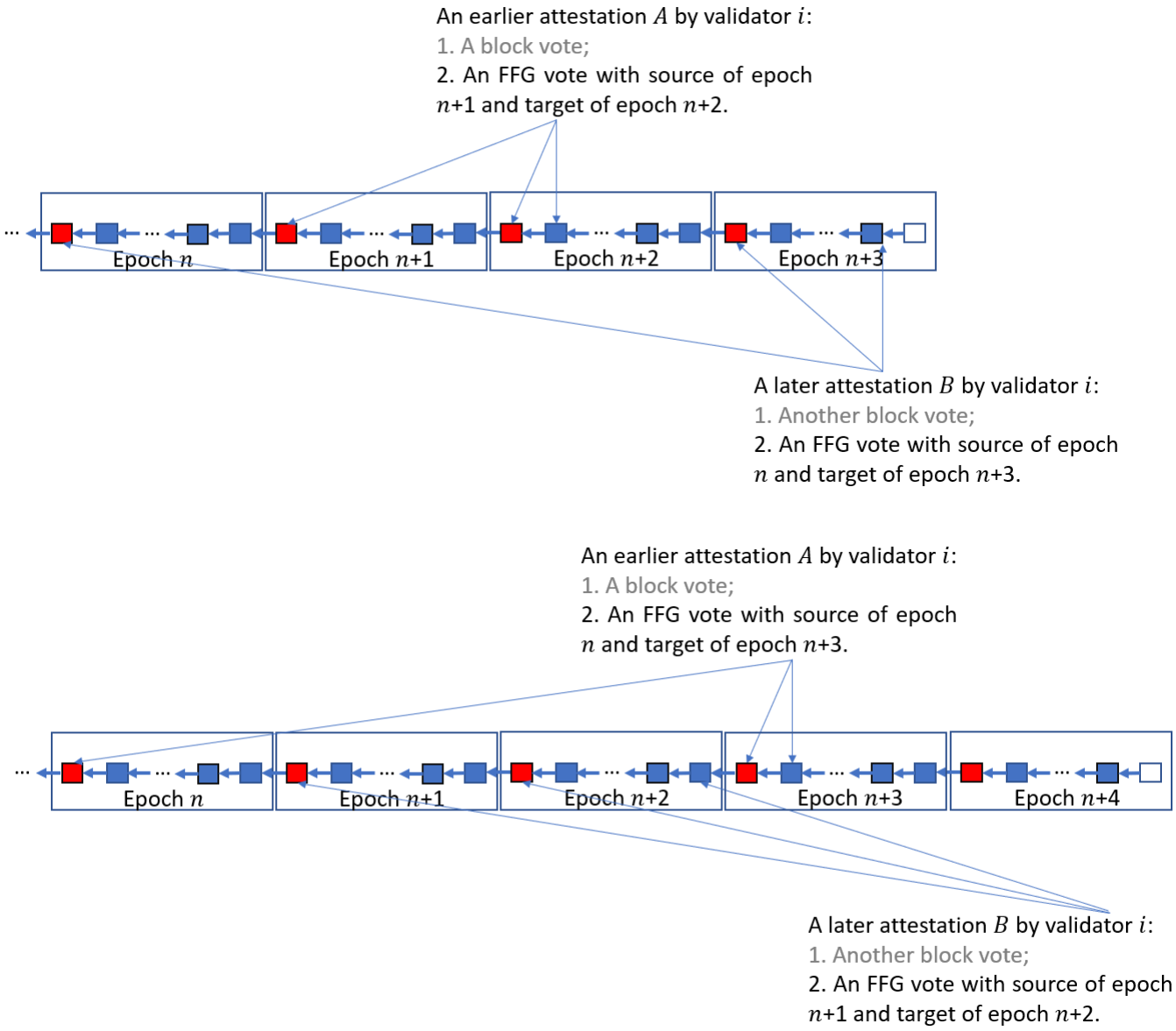

Figure 8: Further illustrations for surround vote

The upper figure (which has appeared in Figure 4) illustrates an FFG vote "surrounding" a previous FFG vote: validator i's attestation A during epoch $n+2$ specifies a source of epoch $n+1$ and a target of epoch $n+2$, while later during epoch $n+3$ the same validator sends a new attestation B which specifies a source of epoch n and target of epoch $n+3$. Then attestation B surrounds attestation A. The lower figure illustrates an FFG vote "surrounded" by a previous FFG vote: validator i's attestation A during epoch $n+3$ specifies a source of epoch n and a target of epoch $n+3$, while later during epoch $n+4$ the same validator sends a new attestation B which specifies a source of epoch $n+1$ and target of epoch $n+2$. Then attestation B is surrounded by attestation A. If we see a validator create such pair of attestations in either order, this validator needs to be slashed. In sum, both surrounding and surrounded attestations count as slashable offenses.

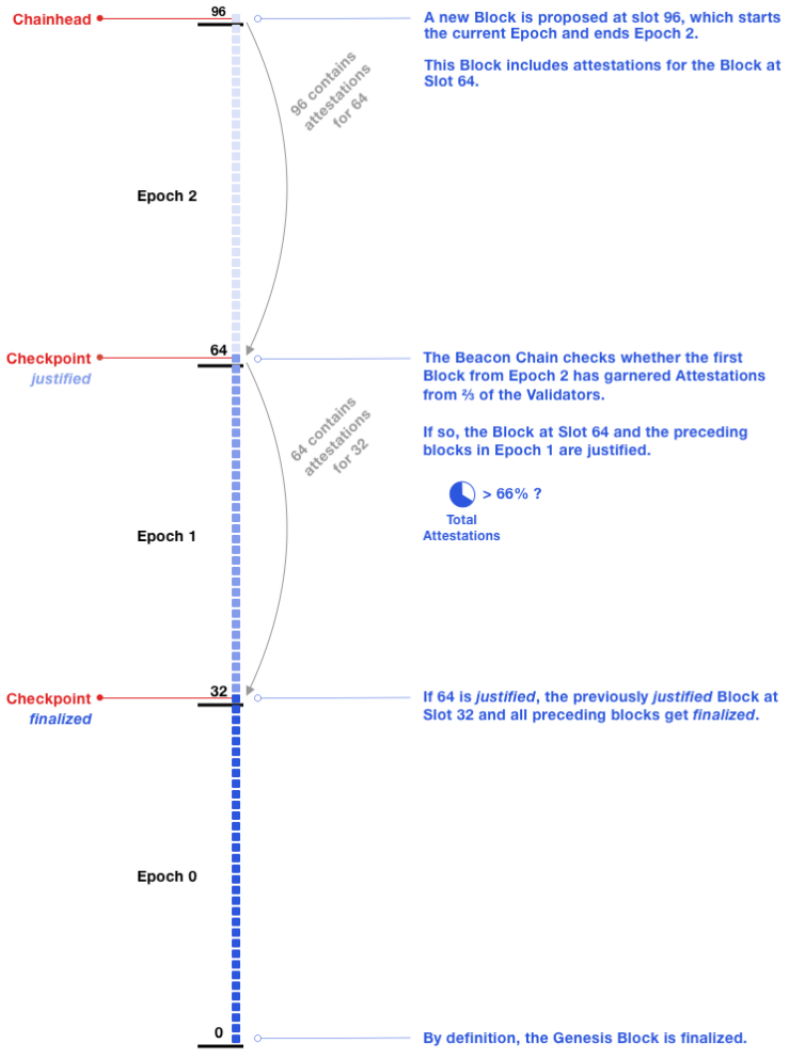

Figure 9: Caption

This figure, borrowed from https://ethos.dev/beacon-chain/, illustrate the justification and finalization of blocks. The genesis block is finalized by default. A checkpoint that has received more than $\frac{2}{3}$ FFG votes from all validators (that is, more than $\frac{2}{3}$ of validators have voted for the same source-target pair with the focal checkpoint being the target) gets justified. The checkpoint for epoch 1 (block 32) is finalized when its immediate next checkpoint, that is, the checkpoint for epoch 2 (block 64) becomes justified.

more than $\frac{2}{3}$ validators have included B as target in their FFG votes. By the pigeon hole principle, at least $\frac{2}{3}+\frac{2}{3}-1=\frac{1}{3}$ validators have included both A and B as targets in their FFG votes. These validators then have committed double votes. (2) If A and B are for different epochs, that is, $e(A) \neq e(B)$. Without loss of generality, assume that A has a smaller epoch number than B, that is, $e(A) < e(B)$. Since A and B conflict, B also conflicts with A's immediately next checkpoint A', which is justified by definition. Then $e(B) > e(A') = e(A) + 1$. Denote C as a justified checkpoint that has the smallest epoch number among the set of all justified checkpoints that conflict with A and have epoch number larger than $e(A)$. Notice that C is well-defined because the set is not empty (for example, B belongs to the set). Then all FFG votes that justify C must have C as target and a source checkpoint D with epoch number smaller than $e(A)$. Therefore, more than $\frac{2}{3}$ validators have included C as target and D as source in their FFG votes, while the finalization of A upon A''s justification means that (not necessarily the same set of) more than $\frac{2}{3}$ validators have included A' as target and A as source in their FFG votes. By the pigeon hole principle, at least $\frac{2}{3}+\frac{2}{3}-1=\frac{1}{3}$ validators have included both source-A/target-A' as well as source-D/target-C in their FFG votes. These validators then have committed surround votes.

C Recorded slashing incidents

Below we present a list of all slashed violations. The list comes from one of the official Beacon chain explore. We separate proposer and attester violations:

(1) proposer violations: The following table lists all slashed proposer violations over time. For each slashing incident, we list the slashed attester's ID, the slashing attester's (the whistleblowing proposer's) ID, the locations of the slashing message (i.e. the block number at which the slashing message is included in the Beacon chain), and the slashable proposal's location (for which block the slashable proposal was made).

slashed proposer	slashing proposer	slashing message location	proposal location
20075	11313	6669	6668
18177	21106	22374	22373

25645	11117	40772	40771
38069	24876	138164	138163
38089	10010	138731	138730
38130	4156	140313	140312
38129	33452	140559	140558
38065	33153	140811	140810
38128	14011	140845	140844
38117	31339	140895	140894
38114	23929	141174	141173
45871	32686	248186	248185
40892	55778	343133	343132
63338	35018	476904	476903
169440	103269	1510279	1510278
21613	49881	1856963	1856962

(2) attester violations: The following table lists all slashed attester violations (including double votes and surround votes) over time. For each slashing incident, we list the slashed attester's ID, the slashing attester's (the whistleblowing proposer's) ID, the locations of the slashing message (i.e. the block number at which the slashing message is included in the Beacon chain), and the slashable vote's content (for which block the slashable vote was cast for).

slashed validator	slashing proposer	slashing message location	attestation content
4259	19030	17112	17090
4100	19030	17112	17090
21574	19030	17112	17090
4110	10689	17206	17078
13869	10689	17206	17064
4102	10055	17188	17082

4086	10055	17188	17084
4390	11111	17184	17072
4451	11398	17227	17073
18249	11398	17227	17073
7635	17942	43920	43917
1644	21844	102389	102388
23241	15703	118136	118135
38061	10063	138194	138163
38105	10063	138194	138163
38113	28390	138221	138163
38091	25012	138770	138730
38106	1406	140924	140894
38148	1406	140924	140894
38116	4617	161508	138730
38058	4617	161508	138163
43843	5959	231183	231180
52866	7079	256812	256809
57976	4966	296756	296752
38038	39883	357060	357059
9143	21353	421395	421394
8320	75282	456892	456891
8275	39945	456894	456893
8250	62836	456895	456894
8239	3935	456896	456895
16509	54711	456945	456944
16491	20174	456949	456948
16523	76042	456958	456957

16479	50948	456959	456958
14415	54596	457006	457005
17377	20912	457452	457451
71654	31047	457549	457548
71676	31047	457549	457548
71401	23641	457551	457550
69812	23641	457551	457550
71665	27582	457552	457549
69884	27582	457552	457549
68648	66049	457553	457552
69358	66049	457553	457552
71614	9093	457554	457552
69895	9093	457554	457552
71690	36015	457555	457552
75715	36015	457555	457549
75162	34679	457556	457549
71603	34679	457556	457555
69391	55472	457557	457555
69817	55472	457557	457556
69716	70705	457558	457557
71664	70705	457558	457555
71671	48837	457559	457555
69732	48837	457559	457558
69772	19756	457560	457559
69841	19756	457560	457559
71708	73025	457561	457552
71673	73025	457561	457559

71699	36460	457562	457561
69866	36460	457562	457561
71663	3443	457563	457562
69809	3443	457563	457562
69756	52088	457564	457562
69388	52088	457564	457562
71646	17397	457565	457564
71593	17397	457565	457563
69786	48737	457566	457561
68593	48737	457566	457563
72499	50677	457567	457558
72074	50677	457567	457555
69717	43018	457568	457567
70044	43018	457568	457565
71709	27546	457570	457561
71672	27546	457570	457567
72084	65347	457571	457551
71718	65347	457571	457565
71714	52332	457572	457553
71734	52332	457572	457559
72081	70964	457573	457563
71744	70964	457573	457564
72421	30689	457574	457559
72082	30689	457574	457552
72491	55133	457575	457556
72493	55133	457575	457556
72503	33972	457576	457551

72496	33972	457576	457552
72508	4276	457577	457556
72511	4276	457577	457555
72807	52592	457578	457557
72674	52592	457578	457559
75172	24678	457579	457549
75045	24678	457579	457557
75204	44836	457580	457557
75212	44836	457580	457552
75723	32278	457581	457556
75711	32278	457581	457548
75699	18965	457582	457563
71743	18965	457582	457555
69873	20457	457585	457551
18989	3395	475787	475786
17395	3395	475787	475786
24696	22116	475789	475786
19001	22116	475789	475788
26278	3523	475790	475789
26201	14712	475793	475792
17164	61098	475794	475793
17304	36059	475796	475795
17228	36059	475796	475795
17140	46477	475797	475796
24703	46477	475797	475795
23179	6765	475798	475795
17291	28872	475802	475799

17189	28872	475802	475799
24528	64094	475803	475802
19017	77243	475805	475804
17232	66279	475807	475806
73292	65386	906882	906880
66420	61303	1003554	1003553
8776	54833	1130722	1130720
3206	7347	1224988	1224987
100190	18368	1232592	1232591
67319	123132	1322722	1322720
119315	104819	1322971	1322969
26447	31691	1348349	1348326
25895	48904	1376001	1376000
25894	138327	1376006	1376000
25893	111788	1379971	1379970
12981	12670	1381594	1381593
78678	128648	1899681	1899680
161751	19758	1956770	1956769
161752	20889	1956780	1956779
9230	173466	1978692	1978690
27442	130871	2008348	2008347
45276	142516	2029833	2029832
26945	204685	2043489	2043488
26988	108925	2176004	2176001
26989	165513	2176738	2176737
26987	103522	2176803	2176801
42708	197363	2332099	2332096

D Undetected slashable misbehavior

We list all incidents within the first 1.75 million blocks. We find that all double proposals have been successfully slashed, yet the same is not true for attestations. We categorize all undetected misbehavior into two groups: double votes and surround votes.

(1) double votes. The following table lists all unslashed double votes, sorted by violators' IDs. For each violation, we list the committing attester's ID, the locations of conflicting votes (i.e. the block number at which each vote is included in the Beacon chain), and the vote content (for which block the conflicting votes were cast for).

attester	vote locations	vote content
237	[1041100, 1041108]	1041099
487	[1041103, 1041108]	1041102
2787	[1041107, 1041108]	1041106
3167	[1041104, 1041108]	1041103
3644	[1267818, 1267822]	1267817
4021	[1054792, 1054796]	1054791
4034	[1267816, 1267822]	1267815
4098	[1041104, 1041108]	1041103
4219	[608067, 608085]	608065
4220	[1041104, 1041108]	1041103
4826	[1041100, 1041108]	1041099
4993	[918915, 918922]	918914
5129	[1054791, 1054791, 1054796]	1054790
5194	[1054788, 1054788, 1054796]	1054787
5228	[1041098, 1041108]	1041097
5327	[1041100, 1041108]	1041099
5837	[1041098, 1041108]	1041097
5942	[1267818, 1267822]	1267817

6146	[1041107, 1041108]	1041106
6274	[1041104, 1041108]	1041103
6531	[1054791, 1054796]	1054790
7080	[1041103, 1041108]	1041102
7131	[1054787, 1054796]	1054786
7378	[1267819, 1267822]	1267818
8195	[1054786, 1054796]	1054785
8510	[1041104, 1041108]	1041103
9648	[1054792, 1054796]	1054791
10898	[1267818, 1267822]	1267817
10914	[1041108, 1041109]	1041107
10915	[1041106, 1041108]	1041105
11608	[1054790, 1054796]	1054789
12127	[1054791, 1054796]	1054790
12666	[1054786, 1054796]	1054785
13283	[1054789, 1054796]	1054788
13574	[1041103, 1041108]	1041102
13601	[1041103, 1041108]	1041102
13620	[1054786, 1054796]	1054785
14429	[1041105, 1041108]	1041104
14539	[1041108, 1041109]	1041107
14730	[1041099, 1041108]	1041098
15755	[1054789, 1054796]	1054788
15963	[1041100, 1041108]	1041099
15987	[1054789, 1054796]	1054788
16849	[1054786, 1054796]	1054785
17111	[1054791, 1054796]	1054790

17413	[1041108, 1041109]	1041107
17436	[1054788, 1054796]	1054787
17491	[1041098, 1041108]	1041097
17604	[918915, 918915, 918916, 918922]	918914
17670	[918915, 918922]	918914
18005	[1054791, 1054796]	1054790
18480	[1041105, 1041108]	1041104
18718	[1041102, 1041108]	1041101
18800	[1054789, 1054796]	1054788
19125	[1041098, 1041108]	1041097
19768	[1041108, 1041109]	1041107
20198	[1041102, 1041102, 1041108]	1041101
20205	[1041100, 1041108]	1041099
20362	[1054789, 1054796]	1054788
20903	[1054788, 1054796]	1054787
20975	[1041108, 1041109, 1041127]	1041107
22255	[1054788, 1054796]	1054787
22277	[1054786, 1054796]	1054785
22520	[1041106, 1041108]	1041105
23700	[1041103, 1041108]	1041102
23705	[1041100, 1041108]	1041099
23989	[1041106, 1041108]	1041105
24067	[1267820, 1267822]	1267819
24163	[1041108, 1041109]	1041107
24632	[918913, 918922]	918912
24787	[1041098, 1041108]	1041097
24817	[1041105, 1041108]	1041104

24939	[918915, 918922]	918914
25130	[1041107, 1041108]	1041106
25132	[1041108, 1041109]	1041107
25359	[988962, 988965, 988976]	988961
25507	[1041098, 1041108]	1041097
25529	[918915, 918922]	918913
25536	[1054788, 1054796]	1054787
25910	[1041105, 1041108]	1041104
26044	[1041105, 1041108]	1041104
26593	[1041099, 1041108]	1041098
26668	[1041107, 1041108]	1041106
26854	[1054787, 1054796]	1054786
26874	[1054787, 1054796]	1054786
26877	[1041103, 1041108]	1041102
26896	[1054789, 1054796]	1054788
27009	[1041100, 1041108]	1041099
27177	[1054786, 1054796]	1054785
27224	[1054789, 1054796]	1054788
27400	[1041103, 1041108]	1041102
27822	[1054790, 1054796]	1054789
27924	[1041100, 1041108]	1041099
28017	[1041107, 1041108]	1041106
28251	[1041099, 1041108]	1041098
28994	[1054789, 1054796]	1054788
29217	[1041104, 1041108]	1041103
29220	[1041098, 1041098, 1041108]	1041097
29545	[1041100, 1041108]	1041099

30078	[1041099, 1041108]	1041098
30095	[1041104, 1041108]	1041103
30218	[1041102, 1041108]	1041101
30389	[1054789, 1054796]	1054788
30423	[918918, 918922]	918917
31024	[1041107, 1041108]	1041106
31101	[1041108, 1041109]	1041107
31537	[1041107, 1041108]	1041106
32456	[1041099, 1041108]	1041098
32620	[1054790, 1054796]	1054789
32770	[1054792, 1054796]	1054791
33096	[1041108, 1041109, 1041117]	1041107
33142	[1041098, 1041108]	1041097
33222	[1041103, 1041108]	1041102
33329	[1054791, 1054796]	1054790
33923	[1054788, 1054788, 1054796]	1054787
34574	[918917, 918918, 918922]	918916
34813	[1054791, 1054796]	1054790
35064	[1054788, 1054788, 1054791, 1054796]	1054787
35746	[1041102, 1041108]	1041101
35787	[1054789, 1054796]	1054788
35897	[1054791, 1054796]	1054790
36238	[1041100, 1041108]	1041099
37098	[1041099, 1041108]	1041098
37435	[1054789, 1054796]	1054788
37607	[1041102, 1041102, 1041102, 1041108]	1041101
37742	[1054791, 1054796]	1054790

39900	[1054792, 1054792, 1054796]	1054791
41238	[1041100, 1041108]	1041099
41376	[1041104, 1041108]	1041103
41755	[1041101, 1041108]	1041100
42021	[918915, 918922]	918914
42061	[1041103, 1041108]	1041102
42377	[1267819, 1267822]	1267818
42850	[1054791, 1054796]	1054790
43098	[1054788, 1054788, 1054796]	1054787
43378	[1054788, 1054796]	1054787
43400	[1054791, 1054791, 1054796]	1054790
43613	[1041098, 1041108]	1041097
43620	[1267817, 1267822]	1267816
43636	[1041100, 1041108]	1041099
44066	[1054786, 1054796]	1054785
44142	[1054790, 1054796]	1054789
44566	[1054791, 1054796]	1054790
44982	[1041106, 1041108]	1041105
45278	[1041108, 1041109]	1041107
46202	[1054786, 1054796]	1054785
47217	[1041104, 1041108]	1041103
47657	[1041103, 1041108]	1041102
47776	[1041100, 1041108]	1041099
48083	[1054787, 1054796]	1054786
48306	[1041107, 1041108]	1041106
48361	[1054791, 1054796]	1054790
48525	[1054791, 1054796]	1054790

48696	[1267818, 1267822]	1267817
48740	[1054789, 1054796]	1054788
48762	[1267819, 1267822]	1267818
48904	[1041106, 1041108]	1041105
48976	[1054790, 1054796]	1054789
49066	[1054791, 1054796]	1054790
49447	[1041108, 1041109]	1041107
49860	[1054789, 1054796]	1054788
49999	[1054787, 1054796]	1054786
50305	[1041103, 1041108]	1041102
50396	[1041098, 1041108]	1041097
51344	[1054790, 1054796]	1054789
51656	[1267819, 1267822]	1267818
51680	[1054788, 1054790, 1054796]	1054787
51877	[1267819, 1267822]	1267818
52090	[1054789, 1054796]	1054788
52451	[1054787, 1054796]	1054786
53715	[1041100, 1041108]	1041099
53813	[1054791, 1054796]	1054790
53851	[1041099, 1041108]	1041098
53896	[1054788, 1054788, 1054791, 1054796]	1054787
54197	[1041108, 1041109]	1041107
54516	[1041099, 1041108]	1041098
54637	[1054787, 1054796]	1054786
55732	[1041098, 1041108]	1041097
55732	[1041098, 1041108]	1041097
55878	[1054787, 1054796]	1054786

55884	[1267818, 1267822]	1267817
56072	[1267816, 1267822]	1267815
56614	[1054790, 1054796]	1054789
56833	[918922, 918938]	918914
57356	[918922, 918938]	918914
58549	[1041102, 1041108]	1041101
58704	[1041103, 1041108]	1041102
58868	[1054791, 1054796]	1054790
59360	[1041099, 1041108]	1041098
59552	[1054786, 1054796]	1054785
59563	[1041105, 1041108]	1041104
59909	[1041099, 1041108, 1041116]	1041098
59988	[1267818, 1267822]	1267817
60533	[1041108, 1041109]	1041107
60652	[1041106, 1041108]	1041105
61264	[1267818, 1267822]	1267817
61378	[1054789, 1054796]	1054788
61503	[1267817, 1267822]	1267816
61782	[1041108, 1041109]	1041107
62170	[1054791, 1054796]	1054790
62593	[1054788, 1054788, 1054791, 1054796]	1054787
62612	[1054787, 1054796]	1054786
63194	[1041101, 1041108]	1041100
63437	[1054786, 1054796]	1054785
63536	[1041099, 1041108]	1041098
63703	[1041105, 1041108]	1041104
63706	[1041099, 1041108]	1041098

63884	[1041101, 1041108]	1041100
64632	[1267816, 1267822]	1267815
65066	[1041104, 1041108]	1041103
65388	[1054791, 1054796]	1054790
65432	[1054786, 1054796]	1054785
65528	[1054788, 1054788, 1054796]	1054787
65558	[1054789, 1054796]	1054788
65665	[1041103, 1041108]	1041102
66303	[1041107, 1041108]	1041106
66391	[1041102, 1041108]	1041101
66411	[1041105, 1041108]	1041104
66588	[1054790, 1054796]	1054789
66611	[1041108, 1041109]	1041107
67040	[1041103, 1041108]	1041102
67555	[1041108, 1041109]	1041107
67683	[1054787, 1054796]	1054786
67792	[1041102, 1041108, 1041113]	1041101
67901	[1041101, 1041108]	1041100
68232	[1041102, 1041108]	1041101
68390	[1041104, 1041108]	1041103
68459	[1054790, 1054796]	1054789
68964	[1267818, 1267822]	1267817
69075	[1041106, 1041108]	1041105
69172	[1054787, 1054796]	1054786
69777	[1041107, 1041108]	1041106
70058	[1041100, 1041108]	1041099
70989	[1267819, 1267822]	1267818

71145	[1041106, 1041108]	1041105
71534	[1054791, 1054796]	1054790
71590	[1054789, 1054796]	1054788
71913	[1054787, 1054796]	1054786
72095	[1267817, 1267822]	1267816
73371	[1054788, 1054788, 1054791, 1054796]	1054787
73462	[1054792, 1054796]	1054791
73518	[1267817, 1267822]	1267816
73754	[1041100, 1041108]	1041099
74304	[1267819, 1267822]	1267818
74366	[1041107, 1041108, 1041128]	1041106
74368	[1267816, 1267822]	1267815
74464	[1041103, 1041108]	1041102
74502	[1054786, 1054796, 1054798, 1054802]	1054785
74969	[1054787, 1054796]	1054786
75015	[1041103, 1041108]	1041102
75085	[1041107, 1041108]	1041106
75651	[1041098, 1041098, 1041108]	1041097
75847	[1054788, 1054796]	1054787
76110	[1267819, 1267822]	1267818
76478	[1054788, 1054796]	1054787
77540	[1054790, 1054796]	1054789
77768	[1041099, 1041108]	1041098
78036	[1054790, 1054796]	1054789
78846	[1054791, 1054796]	1054790
79039	[1041105, 1041108]	1041104
79447	[1041106, 1041108]	1041105

79525	[1041098, 1041108]	1041097
79549	[1041102, 1041108]	1041101
80683	[1041101, 1041108]	1041100
80786	[1041103, 1041108]	1041102
80867	[1041108, 1041109, 1041127]	1041107
80879	[1054788, 1054788, 1054796]	1054787
81284	[1041100, 1041108]	1041099
81385	[1041102, 1041108]	1041101
81541	[1054786, 1054786, 1054788, 1054789, 1054791, 1054792, 1054796]	1054785
82130	[1041098, 1041108]	1041097
82268	[1041100, 1041108]	1041099
82306	[1041099, 1041108, 1041128]	1041098
82769	[1041104, 1041108]	1041103
82948	[1041102, 1041108]	1041101
82963	[1041108, 1041109]	1041107
83048	[1041108, 1041109]	1041107
83114	[1054788, 1054796]	1054787
83431	[1054790, 1054796]	1054789
83812	[1054787, 1054796]	1054786
83923	[1054786, 1054796]	1054785
83923	[1054786, 1054796]	1054785
84677	[1041100, 1041108]	1041099
84782	[1041104, 1041108]	1041103
85787	[1054792, 1054796]	1054791
85951	[1041108, 1041109]	1041107
86425	[1041103, 1041108]	1041102
87736	[1041107, 1041108]	1041106

87777	[1041101, 1041108]	1041100
87863	[1041104, 1041108]	1041103
87982	[1054790, 1054796]	1054789
88298	[1041102, 1041108]	1041101
88654	[1267818, 1267822]	1267817
88726	[988961, 988965]	988960
88832	[1054788, 1054788, 1054796]	1054787
89088	[1041099, 1041108]	1041098
89221	[1041107, 1041108]	1041106
90261	[1054791, 1054796]	1054790
90701	[1041104, 1041108]	1041103
91178	[1041102, 1041108]	1041101
91226	[1041098, 1041108]	1041097
91454	[1041101, 1041108]	1041100
91576	[1041106, 1041108]	1041105
91916	[1054787, 1054796]	1054786
92641	[1041101, 1041108]	1041100
92747	[1054786, 1054796]	1054785
93502	[918918, 918922]	918917
93707	[1041104, 1041108]	1041103
93858	[1041106, 1041108]	1041105
94543	[1041102, 1041108]	1041101
94924	[918913, 918922]	918912
95046	[1054791, 1054796]	1054790
95287	[1041104, 1041108]	1041103
95340	[1041101, 1041108]	1041100
95458	[1041098, 1041108]	1041097

95481	[1041106, 1041108]	1041105
95687	[1041108, 1041109]	1041107
95991	[1041103, 1041108]	1041102
96337	[1041103, 1041108]	1041102
97035	[1041099, 1041108]	1041098
97858	[1054786, 1054796]	1054785
97890	[1041108, 1041109]	1041107
97899	[1267819, 1267822]	1267818
98519	[1054788, 1054796]	1054787
98541	[1041101, 1041108]	1041100
99026	[1041106, 1041108]	1041105
99085	[1041098, 1041108]	1041097
99404	[1041107, 1041108]	1041106
99583	[1041108, 1041109]	1041107
99809	[1041099, 1041108]	1041098
99879	[1041104, 1041108]	1041103
100073	[1054789, 1054796]	1054788
100684	[1054788, 1054796]	1054787
100992	[1054786, 1054796]	1054785
101091	[1267819, 1267822]	1267818
101219	[1041105, 1041108]	1041104
101377	[1041099, 1041108]	1041098
101412	[1041099, 1041108]	1041098
101769	[1041101, 1041108]	1041100
101908	[1267819, 1267822]	1267818
102105	[1041107, 1041108]	1041106
103344	[1041099, 1041108]	1041098

104538	[1041103, 1041108]	1041102
105054	[1054787, 1054796]	1054786
105496	[1267819, 1267820, 1267822, 1267841]	1267818
105510	[1267820, 1267822]	1267819
105787	[1041099, 1041108]	1041098
106142	[1041108, 1041109]	1041107
106581	[1054788, 1054788, 1054796]	1054787
107223	[1267817, 1267822]	1267816
107903	[1041105, 1041108]	1041104
108050	[1054786, 1054796]	1054785
108511	[1054786, 1054796]	1054785
108513	[1267817, 1267822]	1267816
108794	[1041099, 1041108]	1041098
109043	[1041108, 1041109]	1041107
109258	[1041100, 1041108]	1041099
109314	[1041105, 1041108]	1041104
109440	[1054791, 1054796]	1054790
111135	[1041104, 1041108]	1041103
111374	[1041106, 1041108]	1041105
111502	[1041106, 1041108]	1041105
111609	[1041099, 1041108]	1041098
111749	[1054788, 1054788, 1054796]	1054787
111764	[1054787, 1054796]	1054786
111989	[1267819, 1267822]	1267818
112226	[1041108, 1041109]	1041107
113038	[1041100, 1041108]	1041099
113570	[1041105, 1041108, 1041114]	1041104

113751	[1054789, 1054796]	1054788
114633	[1041099, 1041108]	1041098
114847	[1041101, 1041108]	1041100
115560	[1054786, 1054796]	1054785
116145	[1041105, 1041108]	1041104
116961	[1054788, 1054788, 1054789, 1054790, 1054796]	1054787
117028	[1041102, 1041108]	1041101
117455	[1267819, 1267822]	1267818
117566	[1041105, 1041108, 1041118]	1041104
118722	[1041102, 1041108, 1041113]	1041101
118774	[1041098, 1041108]	1041097
119191	[1267819, 1267822]	1267818
119507	[1054789, 1054789, 1054792, 1054796, 1054798, 1054802, 1054805, 1054815]	1054788
120380	[1267818, 1267822]	1267817
120831	[1267818, 1267822]	1267817
120866	[1054791, 1054796]	1054790
121279	[1041108, 1041109]	1041107
121315	[1041098, 1041108]	1041097
121577	[1054789, 1054796]	1054788
122173	[1041099, 1041108]	1041098
122613	[1041106, 1041108]	1041105
122759	[1054787, 1054796]	1054786
122933	[1041100, 1041108]	1041099
123303	[1054790, 1054796]	1054789
123670	[1054786, 1054796]	1054785
124096	[1054788, 1054788, 1054796]	1054787
126276	[1267818, 1267822]	1267817

127043	[1267818, 1267822]	1267817
128021	[1267818, 1267822]	1267817
128827	[1267819, 1267822]	1267818
133070	[1267820, 1267822]	1267819
137005	[1267817, 1267822]	1267816
139563	[1267817, 1267822]	1267816
144716	[1267819, 1267822]	1267818
145763	[1267816, 1267822]	1267815

Some of the incidents above involves orphaned blocks, which may be the reason why they dodged whistle-blower's attention. However, to ensure a well-behaving staking economy, these violations ought to be slashed.

(2) surround votes. The following table lists all unslashed surround votes, sorted by violators' IDs. For each vote within a surround vote violation, we list the committing attester's ID, the location of the vote (i.e. the block number at which the vote is included in the Beacon chain), and the vote content (for which block as well as source and target epoch the vote were cast for).

attester	vote locations	vote content		
		block	source_epoch	target_epoch
4155	918882	918881	28714	28715
4155	918922	918914	28713	28716
4219	608052	608051	19000	19001
4219	608059	608051	19000	19001
4219	608067	608065	18999	19002
4993	918901	918900	28714	28715
4993	918922	918914	28713	28716
6666	988955	988954	30903	30904
6666	988965	988961	30902	30905

7412	918883	918882	28714	28715
7412	918922	918916	28713	28716
9018	988931	988930	30903	30904
9018	988965	988960	30902	30905
10740	918900	918899	28714	28715
10740	918922	918917	28713	28716
11196	918894	918893	28714	28715
11196	918922	918916	28713	28716
13503	918902	918901	28714	28715
13503	918922	918914	28713	28716
14939	918907	918906	28714	28715
14939	918922	918917	28713	28716
15028	988955	988954	30903	30904
15028	988965	988960	30902	30905
16426	988935	988934	30903	30904
16426	988965	988960	30902	30905
17604	918896	918895	28714	28715
17604	918922	918914	28713	28716
17670	918908	918907	28714	28715
17670	918922	918914	28713	28716
19708	918892	918891	28714	28715
19708	918922	918917	28713	28716
20085	918898	918897	28714	28715
20085	918922	918918	28713	28716
20106	918883	918882	28714	28715
20106	918922	918913	28713	28716
20172	918901	918900	28714	28715

20172	918922	918916	28713	28716
24632	918907	918906	28714	28715
24632	918922	918912	28713	28716
24939	918903	918902	28714	28715
24939	918922	918914	28713	28716
25359	988959	988958	30903	30904
25359	988972	988958	30903	30904
25359	988965	988961	30902	30905
25529	918901	918900	28714	28715
25529	918922	918913	28713	28716
25611	918909	918908	28714	28715
25611	918922	918914	28713	28716
29270	918895	918894	28714	28715
29270	918922	918917	28713	28716
30423	918910	918909	28714	28715
30423	918922	918917	28713	28716
32487	918894	918893	28714	28715
32487	918922	918913	28713	28716
34574	918887	918886	28714	28715
34574	918922	918916	28713	28716
37951	918907	918906	28714	28715
37951	918922	918913	28713	28716
39099	918894	918893	28714	28715
39099	918922	918917	28713	28716
39283	918883	918882	28714	28715
39283	918922	918916	28713	28716
40603	918891	918890	28714	28715

40603	918922	918914	28713	28716
41924	918898	918897	28714	28715
41924	918922	918917	28713	28716
42021	918898	918897	28714	28715
42021	918922	918914	28713	28716
47919	918886	918885	28714	28715
47919	918922	918914	28713	28716
49537	608043	608042	19000	19001
49537	608067	608066	18999	19002
50364	608040	608039	19000	19001
50364	608067	608065	18999	19002
51703	918889	918888	28714	28715
51703	918922	918912	28713	28716
52658	918909	918908	28714	28715
52658	918922	918915	28713	28716
52770	918911	918910	28714	28715
52770	918922	918912	28713	28716
52812	918881	918880	28714	28715
52812	918922	918915	28713	28716
54474	988955	988954	30903	30904
54474	988965	988960	30902	30905
56608	988946	988945	30903	30904
56608	988965	988961	30902	30905
56833	918901	918900	28714	28715
56833	918922	918914	28713	28716
57356	918892	918891	28714	28715
57356	918922	918914	28713	28716

57493	988959	988958	30903	30904
57493	988965	988960	30902	30905
58742	918908	918907	28714	28715
58742	918922	918915	28713	28716
59178	988936	988935	30903	30904
59178	988965	988961	30902	30905
60538	988946	988945	30903	30904
60538	988974	988945	30903	30904
60538	988965	988960	30902	30905
61529	918898	918897	28714	28715
61529	918922	918913	28713	28716
62733	918899	918898	28714	28715
62733	918922	918914	28713	28716
63837	918904	918903	28714	28715
63837	918922	918913	28713	28716
64278	918899	918898	28714	28715
64278	918922	918912	28713	28716
64560	918889	918888	28714	28715
64560	918922	918915	28713	28716
65161	918905	918904	28714	28715
65161	918922	918917	28713	28716
68839	918898	918897	28714	28715
68839	918922	918917	28713	28716
72965	918893	918892	28714	28715
72965	918922	918914	28713	28716
84014	918902	918901	28714	28715
84014	918916	918901	28714	28715

84014	918917	918901	28714	28715
84014	918922	918913	28713	28716
85307	918908	918907	28714	28715
85307	918922	918918	28713	28716
86518	918906	918905	28714	28715
86518	918922	918913	28713	28716
87615	988940	988939	30903	30904
87615	988965	988960	30902	30905
88475	608057	608056	19000	19001
88475	608067	608064	18999	19002
88726	988930	988929	30903	30904
88726	988965	988960	30902	30905
88951	918901	918900	28714	28715
88951	918922	918916	28713	28716
93502	918881	918880	28714	28715
93502	918922	918917	28713	28716
94805	988931	988930	30903	30904
94805	988965	988960	30902	30905
94924	918904	918903	28714	28715
94924	918922	918912	28713	28716
96312	918887	918886	28714	28715
96312	918922	918917	28713	28716
97760	918890	918889	28714	28715
97760	918922	918915	28713	28716
105147	918907	918906	28714	28715
105147	918922	918916	28713	28716
106161	918901	918900	28714	28715

106161	918922	918912	28713	28716
111444	918906	918905	28714	28715
111444	918922	918916	28713	28716
112567	918883	918882	28714	28715
112567	918922	918915	28713	28716
114652	918884	918883	28714	28715
114652	918922	918915	28713	28716
115409	918889	918888	28714	28715
115409	918922	918912	28713	28716