Selfish Behavior in the Tezos Proof-of-Stake Protocol

Michael Neuder[†,‡] Daniel J. Moroz[§] Rithvik Rao[¶] David C. Parkes[‖]

Abstract

Proof-of-Stake consensus protocols give rise to complex modeling challenges. We analyze the recently-updated Tezos Proof-of-Stake protocol and demonstrate that, under certain conditions, rational participants are incentivized to behave dishonestly. In doing so, we provide a theoretical analysis of the feasibility and profitability of a block stealing attack that we call *selfish endorsing*, a concrete instance of an attack previously only theoretically considered. We propose and analyze a simple change to the Tezos protocol which significantly reduces the (already small) profitability of this dishonest behavior, and introduce a new delay and reward scheme that is provably secure against length-1 and length-2 selfish endorsing attacks. Our framework provides a template for analyzing other Proof-of-Stake implementations for selfish behavior.

1 Introduction

Blockchain technologies have received significant attention since the release of the Bitcoin protocol in 2008 [21]. Blockchain arose as a solution to the problem of permissionless, decentralized cryptocurrency: how to maintain a global consensus on users' account balances. The Bitcoin system was the first instance of a blockchain, and it addressed this problem by putting every transaction into a globally visible ledger protected by a Proof-of-Work (PoW) mechanism. In this system, each block creator is tasked with assembling a list of valid transactions and doing considerable computational work, and is rewarded with some amount of the native asset, Bitcoin.

Proof-of-Stake (PoS) protocols are widely thought to be the successors of PoW protocols. Both mechanisms run a lottery to select the creator of the next block, and in order to prevent malicious participants from creating many identities to increase their chances of winning, entry into these lotteries must be costly. PoW requires that lottery entrants burn computational cycles in order to join, while PoS requires participants to forego the use of staking capital for a time. While the prospect of wasting energy is sufficient to keep PoW miners following the protocol, PoS protocols require that staked capital be forfeited if miners' behavior is not in line with the protocol. Both systems reward participants proportionally to expenditure. PoW has received significant scrutiny and criticism, and PoS is widely seen as the next step for consensus mechanisms on blockchains.

The three main critiques of PoW systems are the enironmental impact, the inflationary tendencies, and issue of centralization found in many digital currencies. The environmental argument is that PoW has led to significant energy expenditure; by some estimates, the

[†] Dept. of Computer Science, University of Colorado at Boulder. Email: michael.neuder@colorado.edu.

[‡] This work was completed while MN was a visitor at Harvard University.

[§] John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, 02138, USA. Email: dmoroz@g.harvard.edu.

[¶] Harvard College, Cambridge, MA, 02138, USA. Email: rithvikrao@college.harvard.edu

[‖] John A. Paulson School of Engineering and Applied Sciences, Harvard University, 33 Oxford Street, Maxwell Dworkin 229, Cambridge, MA 02138, USA. Email: parkes@eecs.harvard.edu.

annual energy consumption of the Bitcoin network is equivalent to that of Austria [8]. The related 'inflationary' criticism of PoW systems is that they require substantial real world expenditures by miners on hardware and electricity. These miners are in turn compensated by large block rewards that lead to high inflation, and if the rewards are reduced, fewer miners participate and the security of the system degrades. PoS systems do not suffer from these limitations because they do not require much energy expenditure. The 'centralized' argument is that PoW mining power is highly concentrated among a few mining pools. As of Nov 2019, F2Pool owns 18% of the hashpower in the Bitcoin network, and the top four pools combined control more than 50% of the resources [4]. However, achieving any degree of decentralization is non-trivial. Ownership of crytocurrencies is far from decentralized [24], so PoS may not address this issue; indeed, several alternative consensus mechanisms besides PoS have also been presented [2, 20].

While many PoS protocols have been proposed, few are live. PoS protocols have proven difficult to implement and pose novel technical challenges. Ethereum's PoS proposal, Casper [6], is still in development as of November 2019, but EOS [9], Cardano (ADA) [15], BlackCoin [25], Nxt [7], and Tezos [14] are major PoS systems currently running.

The distinguishing feature of Tezos is that it has a built-in upgrading mechanism as part of its protocol. The development of Bitcoin has been slow as few developers want to risk forking the network over a protocol change. In contrast, Tezos hopes to encourage agreement on upgrades by creating a specific venue and timeline for voting on software updates. As of November 2019, each change requires a quorum of participants and over 80% approval to be instantiated [11]. On October 17, 2019, an update labelled 'Babylon' [13, 12] was accepted into the Tezos protocol. Here we analyze a large component of this upgrade: a new consensus mechanism called Emmy$^+$ [19].

2 Related Work

We seek to understand the extent to which rational participants in a particular PoS system can benefit by not behaving according to the protocol.

This question is in the vein of Eyal and Sirer (2013) [10], which demonstrated that miners could earn a higher proportion of rewards in a PoW protocol by 'selfish' mining rather than by following the prescribed 'honest' protocol. Follow-up works include Sapirshtein et al (2016) [23], which found the optimal such policy, Nayak et al (2016) [22], which combined this policy with network attacks, and Kwon et al (2017) [17], which considered the impact of this policy on mining pools.

PoW has been heavily scrutinized, but PoS analysis is still in early days. Recently, Brown-Cohen et al. (2019) showed that complete security in their model of longest-chain PoS protocols is not possible [5]. The dishonest behavior that we call *selfish endorsing* is a real-world instance of the theoretical "Predictable Selfish Mine" attack that appears in their work.

We are not aware of any other academic work formally analyzing the incentives of the Tezos PoS protocol. Nomadic Labs, the team that implemented the recently-updated consensus protocol for Tezos [19], did release a blog post with the results of an incentive analysis [18]. Though we consider the same protocol, their work does not provide an explicit formalization of the model used and the probabilistic analysis performed. We have verified with the authors of the post that our different models achieve similar numerical results when calculating the probability of a profitable attack using the same parameters. In this work, we aim to present the complete derivation of our model and make explicit the methods used to obtain our results.

3 Proof of Stake in Tezos

3.1 The Basics

Tezos implements an optional Delegated Proof of Stake (DPoS) mechanism [14, 11], which is sometimes referred to as Liquid Proof of Stake [1] to distinguish it from the more rigid DPoS implementations [9]. Members of the Tezos consensus layer are called *delegates* and are considered active when they participate in the creation and validation of blocks (passive otherwise). The Tezos unit of account (XTZ) is split into groups of 8,000 tokens called *rolls*, and each delegate has an associated set of rolls. Active delegates participate in a lottery to *bake* and *endorse* a block at every block-height in the chain. Bakers are responsible for including transactions in blocks while endorsers cryptographically sign the "best" (as discussed in Section 3.2) block that they have seen at each height. The baking-and-endorsing-priority lottery is carried out by randomly selecting rolls and giving the next available priority to the owner of that roll, a technique known as *follow-the-Satoshi* [3]. At each height of the chain, a list of bakers is created using the random roll selection process, and the index of a baker in this list determines the priority (as discussed in Section 3.2) with which they can create a block at this height. Additionally a set of 32 endorsers is created for each block-height, but there is no priority list for endorsers and thus each has equal weight. Each draw from the set of rolls is done with replacement so the same delegate may appear many times on the baking priority list and in the set of endorsers. Bakers and endorsers are rewarded based on participation which creates an incentive for delegates to remain active. We now turn our attention to the Babylon update of the protocol.

3.2 The Babylon Upgrade & Emmy$^+$

The new consensus protocol, Emmy$^+$, is distinct from its predecessor, Emmy, in three important ways.

1. A block's validity-time is now a function of the number of endorsements it includes in addition to the priority of the baker. Note that this number of endorsements is *not* the number of delegates who endorse the block itself, but rather the number of endorsements for the previous block that it *includes* (endorsements are simply operations that are heard over the network so including an endorsement is equivalent to including a transaction in a block). In order for a block to be considered valid, its timestamp must differ from the previous block's timestamp by at least $\mathcal{D}$ seconds, where $\mathcal{D}$ is the following function of the baker's priority, p, and the number of endorsements included in the block, e (see *Minimal block delays* in [11]).

$$\mathcal{D}(p, e) = 60 + 40 \cdot p + 8 \cdot \max(24 - e, 0). \tag{1}$$

   Each priority-level a baker is below the highest-priority (0) increases validity-time by 40 seconds, and each endorsement missed below 24 of the 32 increases validity-time by 8 seconds. That is, a block may miss up to 8 endorsements without incurring a time penalty, but each additional missed signature slows validity by 8 seconds. Historical data for the Tezos network shows that the typical block misses few endorsements and that these penalties have been quite rare [16]. If the priority of the baker is 0, a block will typically be baked every 60 seconds.

2. The fork-choice rule was changed. Before this update, the 'canonical' fork was the one that with the most endorsements, a heaviest-chain rule [14], but now the best fork is simply the one with the longest chain from the genesis block. A longest-chain fork-choice rule makes evaluation of branches easier and alleviates a baker's uncertainty of

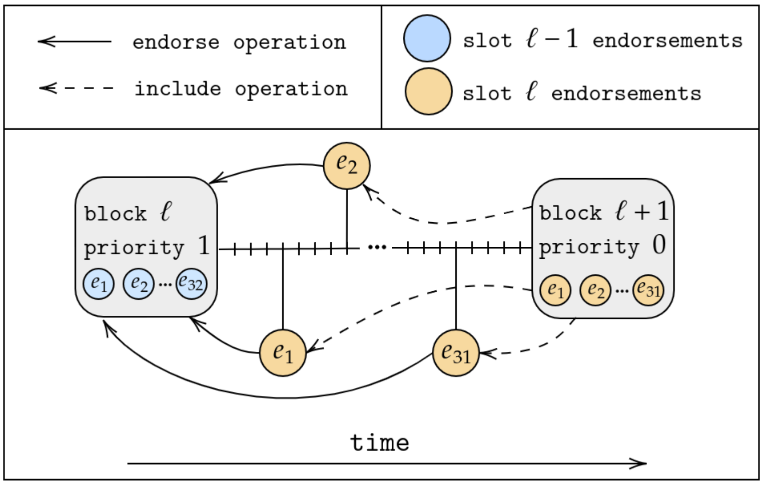

Figure 1: In this scenario, block ℓ will earn the baker a reward of $\mathcal{R}_b(1, 32) = 8$ XTZ, and block $\ell + 1$ will earn the baker $\mathcal{R}_b(0, 31) = 15.9$ XTZ. The endorsements for slot $\ell - 1$ will each earn $\mathcal{R}_e(1) = 1$ XTZ and the endorsements for slot ℓ will each earn $\mathcal{R}_e(0) = 2$ XTZ. Notice that the slot ℓ endorsements still get the full reward when signing a lower priority block, simply because they are included in the 0^{th} priority block at the $\ell + 1$ slot. Before the Babylon upgrade the slot ℓ endorsements would only earn 1 XTZ each.

when to publish blocks to avoid missing out on late endorsements. However, Brown-Cohen et al. show that it also leads to theoretical vulnerabilities [5]. Here we focus on one of these, "Predictable Selfish Mining," and our work highlights a real-world example of this result.

3. The rewards for baking and endorsing blocks were modified. Previously, baking a block earned the delegate a constant reward of 16 XTZ, but now the block rewards are a function of the baker's priority, p, and the number of endorsements for the previous block included, e (see *Rewards* in [11]). Let $\mathcal{R}_b$ be the baking reward. Then:

$$\mathcal{R}_b(p, e) = \frac{16}{p+1}\left(\frac{4}{5} + \frac{1}{5} \cdot \frac{e}{32}\right). \tag{2}$$

The rewards for endorsements also changed subtly. Before Babylon, endorsement rewards were a function of the priority of the block that the endorsement signed, but now they are a function of the priority of the block that *includes* the endorsements. Denote the priority of the baker who baked the block that includes an endorsement p. Then the reward for endorsing, $\mathcal{R}_e$, is

$$\mathcal{R}_e(p) = \frac{2}{p+1}. \tag{3}$$

Figure 1 shows the rewards that the bakers and endorsers would earn under the new reward rules in an arbitrary scenario.

3.3 Notation

We describe baking and endorsing rights in terms of slots that correspond to a specific length of the chain. As implemented, delegates in Tezos see exactly who will have baking and endorsing rights for the next several thousand ($\approx 5 \times 4096$) blocks. Let $\mathcal{X}$ be a rational delegate who is willing to deviate from the protocol described above. For a given slot ℓ,

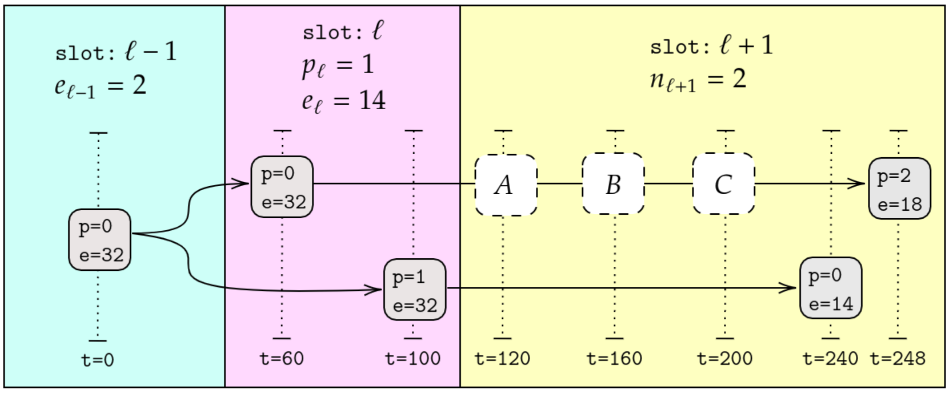

Figure 2: The selfish endorsing attack with $e_{\ell-1} = 2$, $e_\ell = 14$, $p_\ell = 1$, and $n_{\ell+1} = 2$. The top and bottom forks show the next two blocks for the honest and selfish chains respectively. Each block shows the priority of the baker who created it, p, and the number of endorsements that it includes, e. Note that e *isn't* the number of endorsements that it receives, but the number of endorsements for the block baked in the previous slot that it *includes*, thus both blocks created at slot ℓ have $e = 32$ because they both include all 32 endorsements of the block baked at slot $\ell - 1$. The empty blocks labelled A, B, and C correspond to timestamps when the honest network would expect the second block to appear. A and B are empty because $\mathcal{X}$ holds the top two priorities for slot $\ell + 1$. C is empty because the baker with priority 2 for slot $\ell + 1$ is missing $\mathcal{X}$'s 14 endorsements from slot ℓ, which incurs a 48 second penalty.

let p_ℓ be the highest priority and e_ℓ the number of endorsement rights that $\mathcal{X}$ is randomly allocated. Additionally, let $n_{\ell+1}$ denote the number of consecutive top baking priorities given to $\mathcal{X}$ at slot $\ell + 1$ (e.g. if the baking priority list for the $\ell + 1$ slot is $[\mathcal{X}, \mathcal{X}, \texttt{other}, \mathcal{X}, ...]$, then $p_{\ell+1} = 0$ and $n_{\ell+1} = 2$).

4 The Selfish Endorsing Attack

We now give an example of the incentive vulnerability we call *selfish endorsing* that with some probability incentivizes a rational baker $\mathcal{X}$ to ignore the longest-chain rule and create a separate two-block fork faster than the rest of the network. First we show an example where this is possible, which enables a 1-confirmation double-spend, but also show it can be profitable based on block and endorsement rewards alone.

4.1 An Example Attack

Figure 2 shows how this attack can be used by $\mathcal{X}$ to bake a block at slot ℓ that will end up on the final chain despite having second priority, $p_\ell = 1$, and the block baked with priority 0 being published on time. Recall that priorities are zero-indexed with 0 being the highest.

Delegate $\mathcal{X}$ is able to look ahead and calculate when this attack is possible to execute because the baking priorities and endorsing rights are publicly known. We verify that the selfish chain will create two blocks faster than the honest chain using Equation 1. Let $\mathcal{D}_h$

and $\mathcal{D}_s$ be the total time to create both blocks for the honest and selfish chains respectively.

$$\begin{aligned}\mathcal{D}_h &= \mathcal{D}(0,32) + \mathcal{D}(2,18) && (4)\\ &= 248 && (5)\\ \mathcal{D}_s &= \mathcal{D}(1,32) + \mathcal{D}(0,14) && (6)\\ &= 240 && (7)\end{aligned}$$

The combination of having a large share of endorsements for slot ℓ and the two highest priorities for slot $\ell + 1$ allows $\mathcal{X}$ to slow down the honest network by only endorsing a private block at slot ℓ. This enables $\mathcal{X}$ to produce a second valid block before the honest network, and thus create the unique longest chain. Using Equations 2 and 3 we verify that this selfish behavior also results in a greater reward than following the honest protocol, thus demonstrating that this is a profitable deviation in its own right and not just an opportunity for a 1-confirmation double spend. Let $\mathcal{R}_h$ be the total reward earned by $\mathcal{X}$ over the next two blocks while behaving honestly. The labels underneath the expressions indicate the reason each reward is added.

$$\begin{aligned}\mathcal{R}_h &= \underbrace{2 \cdot \mathcal{R}_e(0)}_{e_{\ell-1}=2} + \underbrace{14 \cdot \mathcal{R}_e(0)}_{e_\ell=14} + \underbrace{\mathcal{R}_b(0,32)}_{p_{\ell+1}=0} && (8)\\ &= 4 + 28 + 16 && (9)\\ &= 48 && (10)\end{aligned}$$

Similarly we calculate the total rewards $\mathcal{X}$ earns while selfishly endorsing, denoted $\mathcal{R}_s$.

$$\begin{aligned}\mathcal{R}_s &= \underbrace{2 \cdot \mathcal{R}_e(1)}_{e_{\ell-1}=2} + \underbrace{14 \cdot \mathcal{R}_e(0)}_{e_\ell=14} + \underbrace{\mathcal{R}_b(1,32)}_{p_\ell=1} + \underbrace{\mathcal{R}_b(0,14)}_{p_{\ell+1}=0 \;\&\; e_\ell=14} && (11)\\ &= 2 + 28 + 8 + 14.2 && (12)\\ &= 52.2 && (13)\end{aligned}$$

This demonstrates that the gain in reward from creating a new block at slot ℓ outweighs the loss in reward from the endorsements on slot $\ell - 1$ ending up on a lower priority block and the block baked in slot $\ell + 1$ only including 14 endorsements. Critically, the rewards for the 14 endorsements for slot ℓ (which are essential in slowing down the honest network), do *not* decrease at all because these endorsements are still included on the block baked by $\mathcal{X}$ at slot $\ell + 1$, which has priority 0.

4.2 Feasibility

Figure 2 describes just one instance of a whole family of length-2 selfish endorsing attacks. In order to calculate the probability of any of these attacks happening, we develop a generalized model. Consider tuples of the form $t = (e_{\ell-1},\, e_\ell,\, p_\ell,\, n_{\ell+1})$. We define t as *feasible* if the selfish chain can create two valid blocks faster than the honest chain with this combination of parameters. Figure 2 demonstrates an attack that is feasible with the tuple ($e_{\ell-1} = 2$, $e_\ell = 14$, $p_\ell = 1$, $n_{\ell+1} = 2$), but we want to find all such combinations of parameters with this property. Let $\mathcal{D}_s$ and $\mathcal{D}_h$ be the total time for the selfish and honest networks to create two blocks respectively. Further let $\mathcal{D}_2$ be the difference in the selfish and honest times,

$$\mathcal{D}_2 = \mathcal{D}_s - \mathcal{D}_h. \tag{14}$$

An attack is feasible if $\mathcal{D}_s < \mathcal{D}_h$, which implies $\mathcal{D}_2 < 0$.

Lemma 4.1. $\mathcal{D}_2(p_\ell, e_\ell, n_{\ell+1}) = 40 \cdot (p_\ell - n_{\ell+1}) + 8 \cdot \max(24 - e_\ell, 0) - 8 \cdot \max(e_\ell - 8, 0)$.

Proof. First, we express $\mathcal{D}_h$ in terms of $e_\ell, p_\ell, n_{\ell+1}$ using Equation 1.

$$\begin{aligned}
\mathcal{D}_h(p_\ell, e_\ell, n_{\ell+1}) &= \underbrace{60}_{\mathcal{D}(0,32)} + \underbrace{60 + 40 \cdot n_{\ell+1} + 8 \cdot \max(24 - (32 - e_\ell), 0)}_{\mathcal{D}(n_{\ell+1}, 32 - e_\ell)} && (15)\\
&= 120 + 40 \cdot n_{\ell+1} + 8 \cdot \max(e_\ell - 8, 0) && (16)
\end{aligned}$$

Similarly, we define $\mathcal{D}_s$ as the time it takes for the selfish fork to create two blocks.

$$\begin{aligned}
\mathcal{D}_s(p_\ell, e_\ell, n_{\ell+1}) &= \underbrace{60 + 40 \cdot p_\ell}_{\mathcal{D}(p_\ell,32)} + \underbrace{60 + 8 \cdot \max(24 - e_\ell, 0)}_{\mathcal{D}(0,e_\ell)} && (17)\\
&= 120 + 40 \cdot p_\ell + 8 \cdot \max(24 - e_\ell, 0) && (18)
\end{aligned}$$

Now we solve for $\mathcal{D}_2$.

$$\begin{aligned}
\mathcal{D}_2(p_\ell, e_\ell, n_{\ell+1}) &= \mathcal{D}_s(p_\ell, e_\ell, n_{\ell+1}) - \mathcal{D}_h(p_\ell, e_\ell, n_{\ell+1})\\
&= 120 + 40 \cdot p_\ell + 8 \cdot \max(24 - e_\ell, 0)\\
&\quad - [120 + 40 \cdot n_{\ell+1} + 8 \cdot \max(e_\ell - 8, 0)]\\
&= 40 \cdot (p_\ell - n_{\ell+1}) + 8 \cdot \max(24 - e_\ell, 0) - 8 \cdot \max(e_\ell - 8, 0)
\end{aligned} \qquad (19)$$

□

Now if $\mathcal{D}_2 < 0$ we can assert the feasibility of an attack parameterized with the tuple $t = (e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1})$.

4.3 Profitability

Now we need to similarly parameterize the reward functions. Let $\mathcal{R}_h$ be the reward for delegate $\mathcal{X}$ playing honestly for the next two blocks, and $\mathcal{R}_s$ be the reward for selfish endorsing over that same span. Further let $\mathcal{R}_2$ be the difference in selfish and honest rewards.

$$\mathcal{R}_2 = \mathcal{R}_s - \mathcal{R}_h. \qquad (20)$$

An attack is *profitable* if the amount of rewards that $\mathcal{X}$ receives playing selfishly is greater than that which they would receive playing honestly, or if $\mathcal{R}_2 > 0$. We are slightly abusing notation here in that for delay, the subscripts s and h refer to the delay for $\mathcal{X}$ versus the delay for the rest of the network. In the case of rewards, however, the subscripts both refer to delegate $\mathcal{X}$ and correspond to selfish or honest behavior. Now we define $\mathcal{R}_2$ as a function of the tuple $t = (e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1})$.

Lemma 4.2. $\mathcal{R}_2(p_\ell, e_{\ell-1}, e_\ell) = 16 \cdot \left(\frac{1}{p_\ell+1} + \frac{e_\ell}{160} - \frac{1}{5}\right) + 2e_{\ell-1}\left(\frac{1}{p_\ell+1} - 1\right)$.

Proof. First we focus on the reward for behaving honestly. We must take into account the endorsement rewards and the block rewards for the next two blocks. Using Equations 2 and 3 we define the total reward for honest behavior over then next two blocks, $\mathcal{R}_h$, as

$$\begin{aligned}
\mathcal{R}_h(e_{\ell-1}, e_\ell) &= \underbrace{e_{\ell-1} \cdot \mathcal{R}_e(0)}_{\texttt{slot } \ell \texttt{ rewards}} + \underbrace{e_\ell \cdot \mathcal{R}_e(0) + \mathcal{R}_b(0, 32)}_{\texttt{slot } \ell+1 \texttt{ rewards}} && (21)\\
&= 2e_{\ell-1} + 2e_\ell + 16 && (22)\\
&= 2 \cdot (e_{\ell-1} + e_\ell) + 16 && (23)
\end{aligned}$$

Note that we categorize the rewards for the endorsements under the next slot because that is where they are included. Similarly we calculate the rewards while following the selfish endorsing policy, $\mathcal{R}_s$.

$$\begin{aligned}
\mathcal{R}_s(e_{\ell-1}, e_\ell, p_\ell) &= \underbrace{e_{\ell-1} \cdot \mathcal{R}_e(p_\ell) + \mathcal{R}_b(p_\ell, 32)}_{\texttt{slot } \ell \texttt{ rewards}} + \underbrace{e_\ell \cdot \mathcal{R}_e(0) + \mathcal{R}_b(0, e_\ell)}_{\texttt{slot } \ell{+}1 \texttt{ rewards}} && (24) \\
&= \frac{2}{p_\ell + 1} \cdot e_{\ell-1} + \frac{16}{p_\ell + 1} + 2e_\ell + 16 \cdot \left(\frac{4}{5} + \frac{1}{5} \cdot \frac{e_\ell}{32}\right) && (25) \\
&= 2 \cdot \left(\frac{e_{\ell-1}}{p_\ell + 1} + e_\ell\right) + 16 \cdot \left(\frac{1}{p_\ell + 1} + \frac{4}{5} + \frac{e_\ell}{160}\right) && (26)
\end{aligned}$$

Now we solve for $\mathcal{R}_2$.

$$\begin{aligned}
\mathcal{R}_2(e_{\ell-1}, e_\ell, p_\ell) &= \mathcal{R}_s(e_{\ell-1}, e_\ell, p_\ell) - \mathcal{R}_h(e_{\ell-1}, e_\ell) && (27) \\
&= 2 \cdot \left(\frac{e_{\ell-1}}{p_\ell + 1} + e_\ell - e_{\ell-1} - e_\ell\right) + 16 \cdot \left(\frac{1}{p_\ell + 1} + \frac{4}{5} + \frac{e_\ell}{160} - 1\right) && (28) \\
&= 2e_{\ell-1} \cdot \left(\frac{1}{p_\ell + 1} - 1\right) + 16 \cdot \left(\frac{1}{p_\ell + 1} + \frac{e_\ell}{160} - \frac{1}{5}\right). && (29)
\end{aligned}$$

□

Now if $\mathcal{R}_2 > 0$, we can assert that an attack parameterized by the tuple $t = (e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1})$ is profitable.

4.4 Probability

We now find the probability of a tuple $t = (e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1})$ occurring on the chain. Notice that if α is the percentage of active rolls that $\mathcal{X}$ owns, the probability of $\mathcal{X}$ receiving any priority or endorsement for slot ℓ is α. Then let $\mathcal{P} \sim \text{Geometric}(\alpha)$ be the random variable representing the number of consecutive slots not owned by $\mathcal{X}$. Additionally the probability of $\mathcal{X}$ being allocated the first n priorities in slot $\ell + 1$ also a geometric random variable but with probability $(1 - \alpha)$; let $\mathcal{N} \sim \text{Geometric}(1 - \alpha)$. Lastly, the probability of being allocated e_ℓ endorsement rights for slot ℓ is a binomial random variable with fixed size of 32; let $\mathcal{E} \sim \text{Binomial}(32, \alpha)$. Thus we calculate the probability of tuple $t = (e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1})$ given α.

$$\mathbb{P}[\, t \mid \alpha \,] = \underbrace{(1-\alpha)^{p_\ell}\alpha}_{\mathbb{P}[\mathcal{P}=p_\ell]} \cdot \underbrace{\alpha^{n_{\ell+1}}(1-\alpha)}_{\mathbb{P}[\mathcal{N}=n_{\ell+1}]} \cdot \underbrace{\binom{32}{e_{\ell-1}}\alpha^{e_{\ell-1}}(1-\alpha)^{32-e_{\ell-1}}}_{\mathbb{P}[\mathcal{E}=e_{\ell-1}]} \cdot \underbrace{\binom{32}{e_\ell}\alpha^{e_\ell}(1-\alpha)^{32-e_\ell}}_{\mathbb{P}[\mathcal{E}=e_\ell]} \tag{30}$$

$$= \binom{32}{e_{\ell-1}} \cdot \binom{32}{e_\ell} \cdot \alpha^{n_{\ell+1}+e_{\ell-1}+e_\ell+1} \cdot (1-\alpha)^{65+p_\ell-e_{\ell-1}-e_\ell} \tag{31}$$

4.5 Generalizing

We can now easily calculate the probability of this family of length-2 attacks occurring. Let the set $\mathcal{A}_2$ be all tuples for which the attack is feasible (we will create two blocks faster than the honest network) and profitable (playing selfishly will incur higher rewards than playing honestly). More formally:

$$\begin{aligned}
\mathcal{A}_2 &= \{(e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1}) \mid \mathcal{D}_s < \mathcal{D}_h \wedge \mathcal{R}_h < \mathcal{R}_s\} && (32) \\
&= \{(e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1}) \mid \mathcal{D}_2 < 0 \wedge \mathcal{R}_2 > 0\}. && (33)
\end{aligned}$$

α	$\mathcal{C}\cdot\mathbb{P}[\mathcal{A}_2]$		%	$\mathcal{C}\cdot\mathcal{V}_2$		%
0.1	0.04	0.17	425%	0.09	0.21	233%
0.15	3.88	2.16	56%	7.07	2.02	29%
0.2	33.91	7.70	23%	52.61	6.10	12%
0.25	136.76	12.91	9.4%	175.91	9.00	5.1%
0.3	309.66	12.66	4.1%	324.55	7.92	2.4%
0.35	407.33	8.07	2.0%	361.14	4.60	1.3%
0.4	318.98	3.53	1.1%	254.94	1.85	0.7%

Table 1: The light blue cells represent the results under the current implementation and the orange cells represent the results after our fix is applied. The % column shows the respective ratio of the orange column to the blue column (e.g. 1% means that after the fix, the value of the attack is 1% of what it is currently). We discuss the $\alpha = 0.1$ case in Section 5.

We also want to measure how profitable these attacks are. Let $\mathcal{V}_2$ be the expected increase in reward of the attacks in $\mathcal{A}_2$ (i.e. how much more we make by deviating than by playing honestly).

$$\mathcal{V}_2 = \sum_{t \in \mathcal{A}_2} \mathbb{P}[\, t \mid \alpha \,] \cdot (\mathcal{R}_s(t) - \mathcal{R}_h(t)). \tag{34}$$

Procedure 1 (in Appendix B) demonstrates how these calculations are done, and the blue columns in Table 1 show the results. Let $\mathcal{C} = 365 \cdot 24 \cdot 60$ represent the number of minutes in a year, so $\mathcal{C} \cdot \mathbb{P}[\mathcal{A}_2]$ is the expected number of attacks per year and $\mathcal{C} \cdot \mathcal{V}_2$ is the expected increase in value (in XTZ) for following the selfish policy for a year. This shows the attack is not a serious threat, given that even with 40% of the stake, $\mathcal{X}$ is only expected to earn 254.94 XTC ($\approx$ \$307.23 in November 2019) more than if they had played honestly for the year. Regardless, it is an example of how this type of attack could be formulated against a general longest-chain PoS system.

5 A Heuristic Fix

The profit from this attack can be reduced further by including a simple fix into the protocol. If the rewards for endorsements are reverted to being a function of the block that they endorse instead of the block that includes them, the attacks occur less frequently. The orange columns in Table 1 represent the probability and value of the attack at different levels of α after this fix, and the % column reports the improvement over the status quo. The only exception to this reduction is the case of $\alpha = 0.1$ where we observe that both the probability and the value of the attacks rise as a result of our change; since they remain so low (expected increase in value of 0.21 XTZ $\approx$ \$0.25 over a year), the fix still seems reasonable. The values of the orange columns in Table 1 were calculated using Procedure 1 (in Appendix B), but with the reward function presented in the following lemma. We show the proof in the appendices because it is similar to Lemma 4.2.

Lemma 5.1. *If the endorsement rewards $\mathcal{R}_e$ are now a function of the block that they endorse, p, then $\mathcal{R}_2(p_\ell, e_{\ell-1}, e_\ell) = 16 \cdot \left(\frac{1}{p_\ell+1} + \frac{e_\ell}{160} - \frac{1}{5}\right) + 2e_\ell \left(\frac{1}{p_\ell+1} - 1\right)$.*

Proof. See Appendix A. □

Note that this is *not* a security proof, but rather a heuristic change to decrease the probability and profitability of selfish endorsing for most values of α.

6 Modified Delay and Reward Functions

In contrast to the probabilistic argument above, we now prove that, for particular delay and reward functions, profitable selfish endorsing is not possible. In reality, creating a secure PoS protocol is not as simple as implementing this functionality because there are other long-range forking attacks that must be taken into account. Still, we think it is worth presenting these functions to show that certain PoS systems can be made provably secure against a specific attack vector (length-1 and length-2 selfish endorsing in this case). Let $\mathcal{D}'$ be the new delay function for a block being valid (still a function of the priority of the baker, p, and the number of endorsements it includes, e).

$$\mathcal{D}'(p, e) = 60 + 193 \cdot p + 8 \cdot \max(24 - e, 0) \tag{35}$$

We see that the only new component is the amount of time added for each drop in priority of the baker; in Emmy$^+$ this value is 40 and in $\mathcal{D}'$ it is 193. Now let $\mathcal{R}'_b$ be the modified reward function for a block baked with priority p and including e endorsements. The following reward scheme was proposed by Arthur Breitman as we discussed potential tweaks to the protocol. It maintains an 80 XTZ per block inflation rate, but splits the rewards 40/40 between the baker and the endorsers.

$$\mathcal{R}'_b(p, e) = \frac{5}{4} \cdot \frac{e}{p+1} \tag{36}$$

Now let $\mathcal{R}'_e(p)$ be the reward for an endorsement that is included in a block baked with priority p.

$$\mathcal{R}'_e(p) = \frac{5}{4} \cdot \frac{1}{p+1} \tag{37}$$

We see that if a block is baked with priority 0 and all 32 endorsements are included from the previous block, then the total reward for the block is $\mathcal{R}'_b(0, 32) + 32 \cdot \mathcal{R}'_e(0) = 40 + 40 = 80$ XTZ. Now we prove that length-1 selfish endorsing attacks are not feasible under this new delay schedule and length-2 selfish endorsing attacks are not profitable under this new reward mechanism. Again, this is a useful result in the context of defending against selfish endorsing, but it weakens the system against longer forking attacks. The last section of the analysis done by Nomadic Labs discusses these trade-offs and how the exact constants were selected for Emmy$^+$ [18].

6.1 Security Against Length-1 Selfish Endorsing

The attentive reader may be wondering why we haven't considered single block selfish endorsing attacks until this point. This is due to the following lemma.

Lemma 6.1. *For any tuple $r = (e_{\ell-1}, p_\ell)$ the length-1 selfish endorsing attack is not profitable under Emmy$^+$.*

Proof. See Appendix C. □

When we consider our new delay schedule, we find that a length-1 selfish endorsing attack is never even *feasible*. Assuming the honest network has the highest priority baking rights, let $\mathcal{D}'_{h,1}$ be the time for the honest network to create a single block under the new delay, and $\mathcal{D}'_{s,1}$ be the time for the selfish delegate.

Lemma 6.2. *For any tuple $r = (e_{\ell-1}, p_\ell)$, $\mathcal{D}'_{h,1}(p_\ell, e_{\ell-1}) < \mathcal{D}'_{s,1}(p_\ell, e_{\ell-1})$.*

Proof. In the worst case, the honest network will not receive any endorsements, so the slowest the block creation could be is

$$\mathcal{D}'_{h,1}(p_\ell, e_{\ell-1}) \leq \mathcal{D}'_{h,1}(0,0) = 60 + 8 \cdot 24 \tag{38}$$

$$\leq 252. \tag{39}$$

In the best case for the attacker they own all 32 endorsements and priority of 1 (2^{nd} best) in the block.

$$\mathcal{D}'_{s,1}(p_\ell, e_{\ell-1}) \geq \mathcal{D}'_{s,1}(1,32) = 60 + 193 \tag{40}$$

$$\geq 253. \tag{41}$$

So we have

$$\mathcal{D}'_{h,1} \leq 252 < 253 \leq \mathcal{D}'_{s,1} \implies \mathcal{D}'_{h,1} < \mathcal{D}'_{s,1}. \tag{42}$$

□

6.2 Security Against Length-2 Selfish Endorsing

We will prove that this modified system is secure against length-2 selfish endorsing attacks by demonstrating that they are never profitable. We first need a closed form representation of the rewards $\mathcal{X}$ would receive playing honestly and selfishly under our new reward function, denoted $\mathcal{R}'_h$ and $\mathcal{R}'_s$ respectively. This is derivation is highly similar to that of Lemma 4.2 and thus deferred to the appendices.

Lemma 6.3. *Under the reward policy $\mathcal{R}'_b$ (Equation 36) and $\mathcal{R}'_e$ (Equation 37), the total reward for a rational delegate $\mathcal{X}$ playing honestly over the next two blocks with the tuple $(e_{\ell-1}, e_\ell, p_\ell)$ is*

$$\mathcal{R}'_h(e_{\ell-1}, e_\ell) = 1.25 \cdot (e_{\ell-1} + e_\ell + 32). \tag{43}$$

The total reward for $\mathcal{X}$ to play selfishly over the next two blocks is

$$\mathcal{R}'_s(e_{\ell-1}, e_\ell, p_\ell) = 2.5 \left(\frac{e_{\ell-1}}{p_\ell + 1} + e_\ell \right). \tag{44}$$

Proof. See Appendix D. □

Now we prove that length-2 selfish endorsing under this reward system is never profitable.

Lemma 6.4. *For all tuples in the form $(e_{\ell-1}, e_\ell, p_\ell)$, $\mathcal{R}'_s(e_{\ell-1}, e_\ell, p_\ell) \leq \mathcal{R}'_h(e_{\ell-1}, e_\ell)$.*

Proof. Assume for contradiction that $\mathcal{R}'_s > \mathcal{R}'_h$, which implies

$$2.5 \left(\frac{e_{\ell-1}}{p_\ell + 1} + e_\ell \right) > 1.25 \cdot (e_{\ell-1} + e_\ell + 32). \tag{45}$$

We reduce this algebraically.

$$2 \left(\frac{e_{\ell-1}}{p_\ell + 1} + e_\ell \right) > e_{\ell-1} + e_\ell + 32 \tag{46}$$

$$\frac{2e_{\ell-1}}{p_\ell + 1} + e_\ell - e_{\ell-1} > 32 \tag{47}$$

$$e_{\ell-1} \left(\frac{2}{p_\ell + 1} - 1 \right) + e_\ell > 32 \tag{48}$$

$$e_{\ell-1} \left(\frac{1 - p_\ell}{p_\ell + 1} \right) + e_\ell > 32 \tag{49}$$

Because $p_\ell \geq 1$, we know

$$\frac{1-p_\ell}{p_\ell+1} \leq 0. \tag{50}$$

This along with the fact that $e_{\ell-1} \geq 0$ implies

$$e_{\ell-1}\left(\frac{1-p_\ell}{p_\ell+1}\right) \leq 0. \tag{51}$$

Further, this implies that $e_\ell > 32$ (from line (49)), but we know this is not possible because $e_\ell \in [0, 1, ..., 32]$. By contradiction we conclude that $\mathcal{R}'_s \leq \mathcal{R}'_h$. □

7 Conclusion

This work demonstrates that live PoS systems can be formally analyzed for incentive vulnerabilities. It also serves as a real-world example of the "Predictable Selfish Mine" attack theorized by Brown-Cohen et al [5]. The formalization in our work provides a framework that can be used to check other PoS systems for potential vulnerabilities to selfish behavior by parameterizing a model of time and reward with respect to a specific protocol. We present a modified delay schedule and reward functions that are provably secure against length-1 and length-2 selfish endorsing (though we acknowledge that in practice other attack vectors must also be considered when implementing these mechanisms). While we recognize that, as of November 2019, length-2 selfish endorsing attacks do not seem to be a major threat to the Tezos network, we do demonstrate a simple heuristic modification that reduces the probability and value of many of the attacks by at least an order of magnitude.

There is a wide array of open problems to address in this area, and we see two immediate future directions that build on our work. The first is the goal of a theory of profitable selfish-endorsing attacks beyond length-2. Second, we hope to consider a more general forking attack that on its own would earn a smaller staking reward relative to honest behavior but allows for an attacker to include a double-spend transaction. Both of these questions are considered in [18] but the precise models, derivations, and probabilistic machinery used are not made explicit. We hope that our work serves as a starting point for forthcoming analyses and for a more formal treatment of the security properties of PoS systems.

References

[1] Jacob Arluck. *Liquid Proof-of-Stake*. 2019. URL: https://medium.com/tezos/liquid-proof-of-stake-aec2f7ef1da7 (visited on 11/13/2019).

[2] Arati Baliga. "Understanding blockchain consensus models". In: *Persistent*. 2017.

[3] Iddo Bentov et al. "Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake." In: *IACR Cryptology ePrint Archive* 2014 (2014), p. 452.

[4] Blockchain.com. *Hashrate Distribution. An estimation of hashrate distribution amongst the largest mining pools*. 2019. URL: https://www.blockchain.com/en/pools (visited on 11/13/2019).

[5] Jonah Brown-Cohen et al. "Formal barriers to longest-chain proof-of-stake protocols". In: *Proceedings of the 2019 ACM Conference on Economics and Computation*. ACM. 2019, pp. 459–473.

[6] Vitalik Buterin and Virgil Griffith. "Casper the friendly finality gadget". In: *arXiv preprint arXiv:1710.09437* (2017).

[7] Nxt Community. *Whitepaper:Nxt*. 2019. URL: https://nxtwiki.org/wiki/Whitepaper:Nxt (visited on 11/13/2019).

[8] Digiconomist. *Bitcoin Energy Consumption Index*. 2019. URL: https://digiconomist.net/bitcoin-energy-consumption (visited on 11/13/2019).

[9] EOSIO. *EOS.IO Technical White Paper v2*. 2019. URL: https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md (visited on 11/13/2019).

[10] Ittay Eyal and Emin Gün Sirer. "Majority is not enough: Bitcoin mining is vulnerable". In: *Communications of the ACM* 61.7 (2018), pp. 95–102.

[11] Tezos Foundation. *Proof-of-stake in Tezos*. 2019. URL: https://gitlab.com/tezos/tezos/blob/master/docs/whitedoc/proof_of_stake.rst (visited on 11/13/2019).

[12] Tezos Foundation. *Protocol 005_PsBabyM1 Babylon*. 2019. URL: https://gitlab.com/tezos/tezos/blob/master/docs/protocols/005_babylon.rst (visited on 11/13/2019).

[13] Tezos Foundation. *Update: Week Of 14 October 2019*. 2019. URL: https://tezos.foundation/news/weekly-updates/update-week-of-14-october-2019 (visited on 11/13/2019).

[14] LM Goodman. "Tezos—a self-amending crypto-ledger White paper". In: *URL: https://www. tezos. com/static/papers/white_paper. pdf* (2014).

[15] Aggelos Kiayias et al. "Ouroboros: A provably secure proof-of-stake blockchain protocol". In: *Annual International Cryptology Conference*. Springer. 2017, pp. 357–388.

[16] KidTsunami. *tzstats*. 2019. URL: https://tzstats.com/ (visited on 11/13/2019).

[17] Yujin Kwon et al. "Be selfish and avoid dilemmas: Fork after withholding (faw) attacks on bitcoin". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2017, pp. 195–209.

[18] Nomadic Labs. *Analysis of Emmy+*. 2019. URL: https://blog.nomadic-labs.com/analysis-of-emmy.html (visited on 11/13/2019).

[19] Nomadic Labs. *Emmy+: an improved consensus algorithm*. 2019. URL: https://blog.nomadic-labs.com/emmy-an-improved-consensus-algorithm.html (visited on 11/13/2019).

[20] Du Mingxiao et al. "A review on consensus algorithm of blockchain". In: *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2017, pp. 2567–2572.

[21] Satoshi Nakamoto et al. "Bitcoin: A peer-to-peer electronic cash system". In: (2008).

[22] Kartik Nayak et al. "Stubborn mining: Generalizing selfish mining and combining with an eclipse attack". In: *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2016, pp. 305–320.

[23] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. "Optimal selfish mining strategies in bitcoin". In: *International Conference on Financial Cryptography and Data Security*. Springer. 2016, pp. 515–532.

[24] Balaji S. Srinivasan. *Quantifying Decentralization*. 2017. URL: https://news.earn.com/quantifying-decentralization-e39db233c28e (visited on 11/18/2019).

[25] Pavel Vasin. "Blackcoin's proof-of-stake protocol v2". In: *URL: https://blackcoin. co/blackcoin-pos-protocol-v2-whitepaper. pdf* 71 (2014).

A Proof of Lemma 5.1

Lemma A.1. $\mathcal{R}_2(p_\ell, e_{\ell-1}, e_\ell) = 16 \cdot \left(\frac{1}{p_\ell+1} + \frac{e_\ell}{160} - \frac{1}{5}\right) + 2e_\ell\left(\frac{1}{p_\ell+1} - 1\right)$.

Proof. First we focus on the reward for behaving honestly. We must take into account the endorsement rewards and the block rewards for the next two blocks. We define the total reward for honest behavior over then next two blocks, $\mathcal{R}_h$, be

$$\begin{aligned}
\mathcal{R}_h(e_{\ell-1}, e_\ell) &= \underbrace{e_{\ell-1} \cdot \mathcal{R}_e(0)}_{\texttt{slot } \ell-1 \texttt{ rewards}} + \underbrace{e_\ell \cdot \mathcal{R}_e(0)}_{\texttt{slot } \ell \texttt{ rewards}} + \underbrace{\mathcal{R}_b(0, 32)}_{\texttt{slot } \ell+1 \texttt{ rewards}} &(52)\\
&= 2e_{\ell-1} + 2e_\ell + 16 &(53)\\
&= 2 \cdot (e_{\ell-1} + e_\ell) + 16 &(54)
\end{aligned}$$

Similarly we calculate the rewards while following the selfish endorsing policy, $\mathcal{R}_s$, as

$$\begin{aligned}
\mathcal{R}_s(e_{\ell-1}, e_\ell, p_\ell) &= \underbrace{e_{\ell-1} \cdot \mathcal{R}_e(0)}_{\texttt{slot } \ell-1 \texttt{ rewards}} + \underbrace{\mathcal{R}_b(p_\ell, 32) + e_\ell \cdot \mathcal{R}_e(p_\ell)}_{\texttt{slot } \ell \texttt{ rewards}} + \underbrace{\mathcal{R}_b(0, e_\ell)}_{\texttt{slot } \ell+1 \texttt{ rewards}} &(55)\\
&= 2 \cdot e_{\ell-1} + \frac{16}{p_\ell + 1} + \frac{2}{p_\ell + 1} \cdot e_\ell + 16 \cdot \left(\frac{4}{5} + \frac{1}{5} \cdot \frac{e_\ell}{32}\right) &(56)\\
&= 2 \cdot \left(\frac{e_\ell}{p_\ell + 1} + e_{\ell-1}\right) + 16 \cdot \left(\frac{1}{p_\ell + 1} + \frac{4}{5} + \frac{e_\ell}{160}\right) &(57)
\end{aligned}$$

Now we solve for $\mathcal{R}_2$.

$$\begin{aligned}
\mathcal{R}_2(e_{\ell-1}, e_\ell, p_\ell) &= \mathcal{R}_s(e_{\ell-1}, e_\ell, p_\ell) - \mathcal{R}_h(e_{\ell-1}, e_\ell) &(58)\\
&= 2 \cdot \left(\frac{e_\ell}{p_\ell + 1} + e_{\ell-1} - e_{\ell-1} - e_\ell\right) + 16 \cdot \left(\frac{1}{p_\ell + 1} + \frac{4}{5} + \frac{e_\ell}{160} - 1\right) &(59)\\
&= 2e_\ell \cdot \left(\frac{1}{p_\ell + 1} - 1\right) + 16 \cdot \left(\frac{1}{p_\ell + 1} + \frac{e_\ell}{160} - \frac{1}{5}\right) &(60)
\end{aligned}$$

□

B Procedure 1

Note that × is the Cartesian Product of the lists.

Procedure 1 Find attack probability & value

```
Require: $\alpha$
  E ← [0, 1, ..., 32]
  P ← [1, 2, ..., 20]
  N ← [1, 2, ..., 20]
  totalProb ← 0
  totalValue ← 0
  for $(e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1}) \in$ E × E × P × N do
    if $\mathcal{D}_2 < 0$ and $\mathcal{R}_2 > 0$ then
      currentProb ← $\mathbb{P}[\,(e_{\ell-1}, e_\ell, p_\ell, n_{\ell+1}) \mid \alpha\,]$
      totalProb += currentProb
      totalValue += currentProb $*$ $\mathcal{R}_2$
    end if
  end for
  return (totalProb, totalValue)
```

C Proof of Lemma 6.1

Lemma C.1. *For any tuple $r = (e_{\ell-1}, p_\ell)$ the length-1 selfish endorsing attack is not profitable under Emmy$^+$.*

Proof. First we consider the time it takes the honest network to produce a single block at height ℓ. Denote this value $\mathcal{D}_{h,1}$.

$$\mathcal{D}_{h,1} = 60 + 8 \cdot \max(24 - (32 - e_{\ell-1}), 0) \tag{61}$$
$$= 60 + 8 \cdot \max(e_{\ell-1} - 8, 0) \tag{62}$$

Now we find the time it takes for the selfish delegate to create a block, $\mathcal{D}_{s,1}$.

$$\mathcal{D}_{s,1} = 60 + 40p_{\ell-1} + \max(24 - e_{\ell-1}, 0) \tag{63}$$

Consider that the best case scenario for the attacker is having $p_\ell = 1$.

$$\mathcal{D}_{s,1} = 100 + \max(24 - e_{\ell-1}, 0) \tag{64}$$

Now we find the amount of endorsements required for the selfish delegate to produce a valid block faster than the honest network to be $e_{\ell-1} = 19$. We verify this with the following calculations.

$$\mathcal{D}_{s,1}(19) = 100 + 8 \cdot 5 \tag{65}$$
$$= 140 \tag{66}$$
$$\mathcal{D}_{h,1}(19) = 60 + 8 \cdot 11 \tag{67}$$
$$= 148 \tag{68}$$

So now we know that $e_{\ell-1} = 19$ is the best case scenario for the attack being feasible. Additionally we know that the reward for playing honestly for this block is $\mathcal{R}_{h,1} = 2e_{\ell-1}$ because we will get all our endorsement rewards, and the reward for playing selfishly will be

$$\mathcal{R}_{s,1} = \underbrace{8 \cdot \left(\frac{4}{5} + \frac{1}{5} \cdot \frac{e_{\ell-1}}{32}\right)}_{\mathcal{R}_b(1,e_{\ell-1})} + e_{\ell-1} \cdot \underbrace{1}_{\mathcal{R}_e(1)} \tag{69}$$

Plugging in $e_{\ell-1} = 19$ we have

$$\mathcal{R}_{h,1} = 19 \cdot 2 = 38 \tag{70}$$
$$\mathcal{R}_{s,1} = 8 \cdot \left(\frac{4}{5} + \frac{1}{5} \cdot 19/32\right) + 19 = 26.35 \tag{71}$$

So even for the smallest value of $e_{\ell-1}$ that makes the attack feasible, the profit gained from creating a new block does not outweigh the profit lost for the endorsements ending up on a worse block. So because $\mathcal{R}_{h,1} > \mathcal{R}_{s,1}$, single block selfish endorsing attacks are not profitable under Emmy$^+$. □

D Proof of Lemma 6.3

Lemma D.1. *Under the reward policy $\mathcal{R}'_b$ (Equation 36) and $\mathcal{R}'_e$ (Equation 37), the total reward for a rational delegate $\mathcal{X}$ playing honestly over the next two blocks with the randomly allocated tuple $(e_{\ell-1}, e_\ell, p_\ell)$ is*

$$\mathcal{R}'_h(e_{\ell-1}, e_\ell) = 1.25 \cdot (e_{\ell-1} + e_\ell + 32). \tag{72}$$

The total reward for $\mathcal{X}$ to play selfishly over the next two blocks is

$$\mathcal{R}'_s(e_{\ell-1}, e_\ell, p_\ell) = 2.5\left(\frac{e_{\ell-1}}{p_\ell+1} + e_\ell\right). \tag{73}$$

Proof. This derivation is very similar to 4.2, but with the new reward functions.

$$\mathcal{R}'_h(e_{\ell-1}, e_\ell) = \underbrace{e_{\ell-1}\cdot\mathcal{R}'_e(0)}_{\texttt{slot } \ell \texttt{ rewards}} + \underbrace{e_\ell\cdot\mathcal{R}'_e(0) + \mathcal{R}'_b(0,32)}_{\texttt{slot } \ell+1 \texttt{ rewards}} \tag{74}$$

$$= 1.25e_{\ell-1} + 1.25e_\ell + 1.25(32) \tag{75}$$

$$= 1.25\cdot(e_{\ell-1} + e_\ell + 32) \tag{76}$$

$$\mathcal{R}'_s(e_{\ell-1}, e_\ell, p_\ell) = \underbrace{e_{\ell-1}\cdot\mathcal{R}'_e(p_\ell) + \mathcal{R}'_b(p_\ell,32)}_{\texttt{slot } \ell \texttt{ rewards}} + \underbrace{e_\ell\cdot\mathcal{R}'_e(0) + \mathcal{R}'_b(0,e_\ell)}_{\texttt{slot } \ell+1 \texttt{ rewards}} \tag{77}$$

$$= \frac{1.25}{p_\ell+1}\cdot e_{\ell-1} + \frac{1.25e_{\ell-1}}{p_\ell+1} + 1.25e_\ell + 1.25e_\ell \tag{78}$$

$$= \frac{2.5e_{\ell-1}}{p_\ell+1} + 2.5e_\ell \tag{79}$$

$$= 2.5\left(\frac{e_{\ell-1}}{p_\ell+1} + e_\ell\right) \tag{80}$$

□