consensus seminar

ertosns

2023/6/7

1 Overview

The DarkFi blockchain is proof of stake privacy focused, with high incentive monetary policy.

1.1 Blockchain

Blockchain $\mathbb{C}_{\lessdot \rtimes}$ is a series of epochs: it's a tree of chains, C^1, C^2, $\ldots$, C^n, the chain ending in a single leader per slot signals finalization.

Each participant stores it's own local view of the Blockchain C_{loc} is a sequence of blocks B_i (i¿0), where each $B \in C_{loc}$

$$B = (tx_{lead}, st)$$

$$tx_{lead} = (LEAD, header, txs, stx_{proof})$$

LEAD is a magic word, header is a metadata, and txs is a vector of transaction hash $stx_{proof} = (cm_{\prime c}, sn_c, ep, sl, \rho, h, \pi)$ the Block's st is the block data, and h is the hash of that data. the commitment of the newly created coin is: $(cm_{c_2}, r_{c_2}) = COMM(pk^{COIN}||\tau||v_c||\rho_{c_2})$, τ is slot timestamp, or index. sn_c is the coin's serial number revealed to spend the coin.

$$sn_c = PRF^{sn}_{root^{COIN}_{sk}}(\rho_c)$$

$$\rho = \eta^{sk^{COIN}_{sl}}$$

η is on-chain random oracle , ρ id derived randomness from η. π is the NIZK proof of the LEAD statement.

1.2 st transactions

the blockchain view is a chain of blocks, each block $B_j = (tx_{lead}, st)$, while st being the merkle tree structure of the validated transactions received through the network, that money transfer, and lead transactions.

1.3 LEAD statement

for $x = (cm_{c_2}, sn_{c_1}, \eta, sl, \rho, h, ptr, \mu_\rho, \mu_y, root)$, and
$w = (path, root_{sk^{COIN}}, path_{sk^{COIN}}, \tau_c, \rho_c, r_{c_1}, v, r_{c_2})$ for tuple $(x, w) \in L_{lead}$ iff:

$pk^{COIN} = PRF^{pk}_{root^{COIN}_{sk}}(\tau_c)$.

$\rho_{c_2} = PRF^{evl}_{root^{COIN}_{sk_{c_1}}}(\rho_{c_1})$. note here the nonce of the new coin is deterministically driven from the nonce of the old coin, this works as resistance mechanism to allow the same coin to be eligible for leadership more than once in the same epoch.

$\forall i \in \{1, 2\} : DeComm(cm_{c_i}, pk^{COIN}||v||\rho_{c_i}, r_{c_i}) = T$.

path is a valid Merkle tree path to cm_{c_1} in the tree with the root root.

$path_{sk^{COIN}}$ is a valid path to a leaf at position $sl - \tau_c$ in a tree with a root $root^{COIN}_{sk}$.

$sn_{c_1} = PRF^{sn}_{root^{COIN}_{sk}}(\rho_{c_1})$

$y = \mu_y^{root^{COIN}_{sk_{c_1}}||\rho_c}$

$\rho = \mu_\rho^{root^{COIN}_{sk_{c_1}}||\rho_c}$

$y < T(v)$ note that this process involves burning old coin c_1, minting new c_2 of the same value + reward.

1.3.1 validation rules

validation of proposed lead proof as follows:

- slot index is less than current slot index, epoch index is less than or equal current epoch index.
- proposal extend from valid fork chain
- proposal extend highest ranking block (see 3.2)
- transactions doesn't exceed max limit
- signature is valid based off producer public key
- verify block hash
- verify previous block hash
- public inputs μ_y, μ_{rho} are derived from on-chain VRF output η, and current slot, and block producer signature, and public key.
- public inputs of target 2-term approximations σ_1, σ_2 are valid given total network stake and controller parameters

- the competing coin nullifier isn't published before to protect against double spening, before burning the coin.
- burnt coin's commitment rooted by commitments merkle tree root.
- verify block transactions

1.4 Epoch

An epoch is a vector of blocks. Some of the blocks might be empty if there is no winnig leader. tokens in stake are constant during the epoch.

1.5 Leader selection

At the onset of each slot each stakeholder needs to verify if it's the weighted random leader for this slot.

$$y < T_i$$

check if the random y output is less than some threshold

This statement might hold true for zero or more stakeholders, thus we might end up with multiple leaders for a slot, and other times no leader. Also note that no node would know the leader identity or how many leaders are there for the slot, until it receives a signed block with a proof claiming to be a leader.

sid is block id

$$\phi_f = 1 - (1 - f)^{\alpha_i}$$

$$T_i = L\phi_f(\alpha_i^j)$$

Note that $\phi_f(1) = f$, **f**: the active slot coefficient is the probability that a party holding all the stake will be selected to be a leader. Stakeholder is selected as leader for slot j with probability $\phi_f(\alpha_i)$, α_i is U_i relative stake.

1.6 automating f tuning

the stable consensus token supply is maintained by the help of discrete PID controller, that maintain stabilized occurance of single leader per slot.

1.6.1 control lottery f tunning paramter

$$f[k] = f[k-1] + K_1 e[k] + K_2 e[k-1] + K_3 e[k-2]$$

with $k_1 = k_p + K_i + K_d$, $k_2 = -K_p - 2K_d$, $k_3 = K_d$, and e is the error function.

1.7 target T n-term approximation

target function is approximated to avoid use of power, and division in zk, since no function in the family of functions that have independent aggregation property achieve avoid it (see appendix).

1.7.1 target function

target fuction T:

$$T = L * \phi(\sigma) = L * (1 - (1 - f)^{\sigma})$$

σ is relative stake. f is tuning parameter, or the probability of winning have all the stake L is field length

1.8 $\phi(\sigma)$ approximation

$$\begin{aligned}
\phi(\sigma) &= 1 - (1 - f)^{\sigma} \\
&= 1 - e^{\sigma ln(1-f)} \\
&= 1 - (1 + \sum_{n=1}^{\infty} \frac{(\sigma ln(1-f))^n}{n!}) \\
\sigma &= \frac{s}{\Sigma}
\end{aligned}$$

s is stake, and Σ is total stake.

1.8.1 target T n term approximation

$$\begin{aligned}
k &= Lln(1-f)^1 \\
k^{'n} &= Lln(1-f)^n \\
T &= -[k\sigma + \frac{k^{''}}{2!}\sigma^2 + \cdots + \frac{k^{'n}}{n!}\sigma^n] \\
&= -[\frac{k}{\Sigma}s + \frac{k^{''}}{\Sigma^2 2!}s^2 + \cdots + \frac{k^{'n}}{\Sigma^n n!}s^n]
\end{aligned}$$

1.8.2 comparison of original target to approximation

1.9 Linear family functions

In the previous leader selection function, it has the unique property of independent aggregation of the stakes, meaning the property of a leader winning leadership with stakes σ is independent of whether the stakeholder would act as a pool of stakes, or distributed stakes on competing coins. "one minus the probability" of winning leadership with aggregated stakes is $1 - \phi(\sum_i \sigma_i) = 1 - (1 + (1 - f)^{\sigma_i}) = -(1 - f)^{\sum_i \sigma_i}$, the joint "one minus probability" of all the

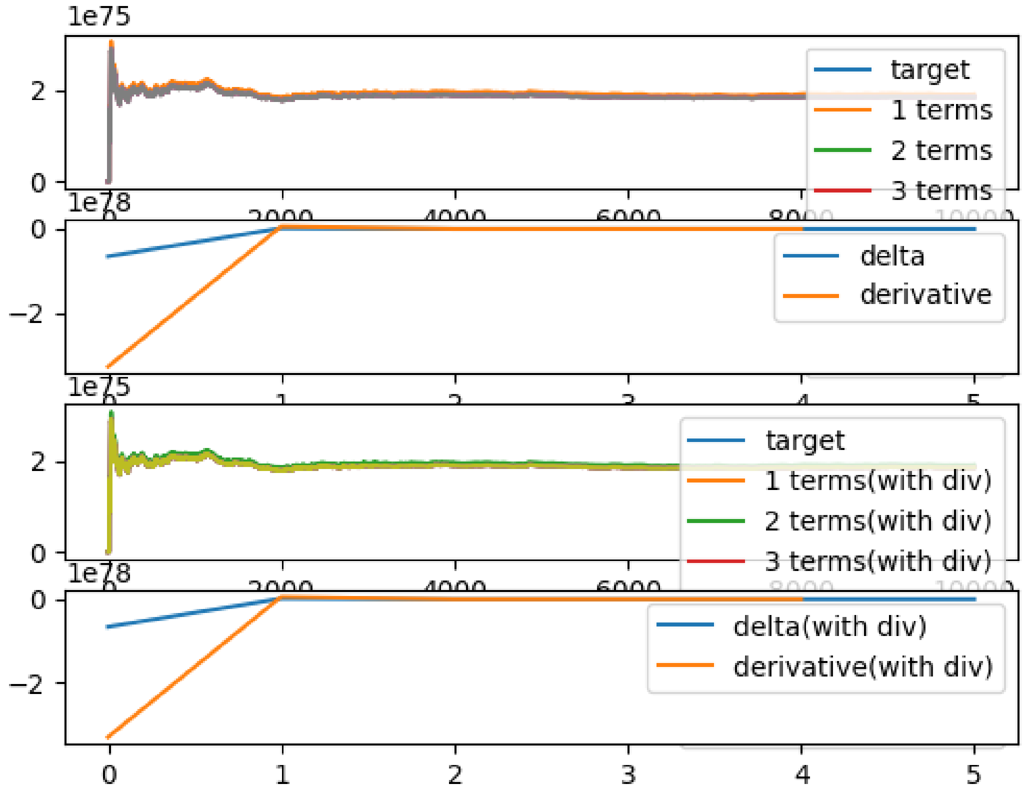

Figure 1: approximation comparison to the original

stakes (each with probability $\phi(\sigma_i)$) winning aggregated winning the leadership $\prod_i^n(1-\phi(\sigma_i)) = -(1-f)^{\sum_i(\sigma_i)}$ thus:

$$1-\phi(\sum_i \sigma_i) = \prod_i^n (1-\phi(\sigma_i))$$

A non-exponential linear leader selection can be:

$$y < T$$
$$y = 2^l k \mid 0 \leq k \leq 1$$
$$T = 2^l \phi(v)$$
$$\phi(v) = \frac{1}{v_{max+} + c} v \mid c \in \mathbb{Z}$$

1.9.1 Dependent aggregation

Linear leader selection has the dependent aggregation property, meaning it's favorable to compete in pools with sum of the stakes over aggregated stakes of distributed stakes:

$$\phi(\sum_i \sigma_i) > \prod_i^n \sigma_i$$
$$\sum_i \sigma_i > (\frac{1}{v_{max} + c})^{n-1} v_1 v_2 \dots v_n$$

let's assume the stakes are divided to stakes of value $\sigma_i = 1$ for $\Sigma > 1 \in \mathbb{Z}$, $\sum_i \sigma_i = V$

$$V > (\frac{1}{v_{max} + c})^{n-1}$$

note that $(\frac{1}{v_{max}+c})^{n-1} < 1, V > 1$, thus competing with single coin of the sum of stakes held by the stakeholder is favorable.

1.9.2 Scalar linear aggregation dependent leader selection

A target function T with scalar coefficients can be formalized as

$$T = 2^l k \phi(\Sigma) = 2^l (\frac{1}{v_{max} + c}) \Sigma$$

let's assume $v_{max} = 2^v$, and $c = 0$ then:

$$T = 2^l k \phi(\Sigma) = 2^{l-v} \Sigma$$

then the lead statement is

$$y < 2^{l-v} \Sigma$$

for example for a group order or l= 24 bits, and maximum value of $v_{max} = 2^{10}$, then lead statement:

$$y < 2^{14} \Sigma$$

1.9.3 Competing max value coins

For a stakeholder with nv_{max} absolute stake, $\mid n \in \mathbb{Z}$ it's advantageous for the stakeholder to distribute stakes on n competing coins.

1.9.4 Inverse functions

Inverse lead selection functions doesn't require maximum stake, most suitable for absolute stake, it has the disadvantage that it's inflating with increasing rate as time goes on, but it can be function of the inverse of the slot to control the increasing frequency of winning leadership.

1.9.5 Leader selection without maximum stake upper limit

The inverse leader selection without maximum stake value can be $\phi(v) = \frac{v}{v+c} \mid c > 1$ and inversely proportional with probability of winning leadership, let it be called leadership coefficient.

1.9.6 Decaying linear leader selection

As the time goes one, and stakes increase, this means the combined stakes of all stakeholders increases the probability of winning leadership in next slots leading to more leaders at a single slot, to maintain, or to be more general to control this frequency of leaders per slot, c (the leadership coefficient) need to be function of the slot sl, i.e $c(sl) = \frac{sl}{R}$ where R is epoch size (number of slots in epoch).

1.9.7 Pairing leader selection independent aggregation function

The only family of functions $\phi(\alpha)$ that are isomorphic to summation on multiplication $\phi(\alpha_1 + \alpha_2) = \phi(\alpha_1)\phi(\alpha_2)$(having the independent aggregation property) is the exponential function, and since it's impossible to implement in plonk, a re-formalization of the lead statement using pairing that is isomorphic to summation on multiplication is an option.

Let's assume ϕ is isomorphic function between multiplication and addition, $\phi(\alpha) = \phi(\frac{\alpha}{2})\phi(\frac{\alpha}{2}) = \phi(\frac{\alpha}{2})^2$, thus:

$$\phi(\alpha) = \underbrace{\phi(1)\dots\phi(1)}_{\alpha} = \phi(1)^\alpha$$

then the only family of functions $\phi : \mathbb{R} \to \mathbb{R}$ satisfying this is the exponential function

$$\phi(\alpha) = c^\alpha \mid c \in \mathbb{R}$$

1.9.8 no solution for the lead statement parameters, and constants S, f, α defined over group of integers.

assume there is a solution for the lead statement parameters and constants S, f, α defined over group of integers. for the statement $y < T$,

$$T = L\phi_{max}\phi(\alpha) = S\phi(\alpha)$$

$$S = ord(G)\phi_{max}\phi(\alpha)$$

such that S inZ $\phi_{max} = \phi(\alpha_{max})$ where α_{max} is the maximum stake value being 2^{64}, following from the previous proof that the family of function haveing independent aggregation property is the exponential function f^{α}, and $f \in Z | f > 1$, the smallest value satisfying f is $f = 2$, then

$$\phi_{max} = 2^{2^{64}}$$

note that since $ord(G) << \phi_{max}$ thus $S << 1$, contradiction.

1.9.9 Leaky non-resettable beacon

Built on top of globally synchronized clock, that leaks the nonce η of the next epoch a head of time (thus called leaky), non-resettable in the sense that the random nonce is deterministic at slot s, while assuring security against adversary controlling some stakeholders.

For an epoch j, the nonce η_j is calculated by hash function H, as:

$$\eta_j = H(\eta_{j-1}||j||v)$$

v is the concatenation of the value ρ in all blocks from the beginning of epoch e_{i-1} to the slot with timestamp up to $(j-2)R + \frac{16k}{1+\epsilon}$, note that k is a persistence security parameter, R is the epoch length in terms of slots.

1.10 toward better decentralization in ouroboros

the randomization of the leader selection at each slot is hinged on the random y, μ_y, ρ_c, those three values are dervied from η, and root of the secret keys, the root of the secret keys for each stakeholder can be sampled, and derived beforehand, but η is a response to global random oracle query, so it's security is hinged on *centralized global random node.*

1.10.1 solution

to break this centeralization, a decentralized emulation of G_{ro} functionality for calculation of: $\eta_i = PRF^{G_{ro}}_{\eta_{i-1}}(\psi)$

$$\psi = hash(tx_0^{ep})$$

$$\eta_0 = hash(\text{"let there be dark!"})$$

note that first transaction in the block, is the proof transaction.

2 on-chain true random oracle

2.1 democratized random lottery seed

at slot i, randomness η_i in the y variable in the leader election comparison, is the output of VRF.

$$\eta_i = VRF(coin_{sk}, \eta_{i-1}||slot_i)$$

which is publicly verifiable through $verify(\eta_{i-1}||slot_{id}, coin_{pk})$, where $coin_{pk}$ is constrained as public input, and keypair $(coin_{sk}, coin_{pk})$ is protected against grinding attack by published nullifier in the same contract, has advantage over ouroboros, which reduce the grinding effect through multiple queries for the random oracle for favoring high probability of winning by limiting the number of queries to the random oracle. [1]

2.2 grinding η, and delayed stake contract

although randomness from the lead stakeholder provider is decentralized, doesn't reveal identity of the provider, more robust than limited access global random oracle, it's deterministic, predictable, meaning, even with time-locked contracts (see 5), there is a window between un-stake, and stake contracts, or for new adversarial stakeholder, it's possible to target winning at $slot_{id}$, by trying different key pairs, and picking secret key corresponding with lowest y possible. that attack can be prevented by delaying reward proposals from newly staked coins by N slots, N be at least 1, N being a security parameter.

2.3 omerta attack

although selecting key pairs corresponding of η with lowest corresponding y using public on-chain data is fixed, it's still possible to make off-chain agreements, for example at slot i if the contract execution is delayed N slots, it's possible through N agreements with winners of slots i to i+N, and since calculation of η is predictable, it's possible to pick elect key pair corresponding to lowest y at slot i+N+1 in the future, early at i slot. note that the off-chain agreement is done with probabilistic winners, but once this attack is done once (even if possible with low probability), the adversaries will have high leverage on winning, by mining lowest possible y.

the attack can be fixed, by introducing unpredictable random parameter:

$$\eta_i = VRF(coin_{sk}, \eta_{i-1}||hash(block_{i-2})||slot_i)$$

note! secret key is constrained 1 slot ahead (assume $N = 1$ for simplicity), and grinding isn't possible since the staked coin is restricted from reward proposals for 1 slot, and η_{i-1} isn't known yet, omerta attack isn't possible since $hash(block_{i-2}))$ isn't known ahead of time, also η_{i-1}, and $hash(block_{i-2})$ are phased out to prevent lead at slot *i-1* from grinding $hash(block_{i-1})$ through different transactions permutations.

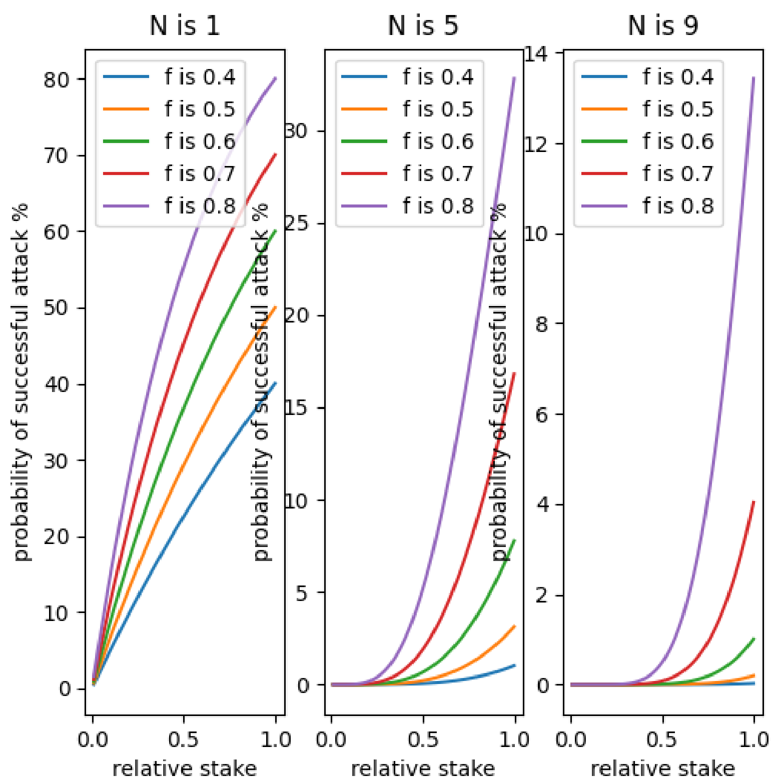

Figure 2: probability of attack for different security parameter

2.4 perpetual lead with predictable block hash

slots are divided into N+2 cycles, (N is security parameter $N \geq 1$), adversary assume winning lottery in the consecutive slots i to i+N, with certain probability, then winning leadership at slot i+N+1 becomes highly probable, only the first block at slot i include a single stake transaction (note staked DRK execution is delayed to slot i+N+1), rest of blocks i+1 to i+N are left with empty transactions, then predicting η_i to η_{i+N+1} is possible, and thus grinding attack can be executed.

if we plot that probability of successful attack, for N (security parameter) delayed slots, function of relative stake (see figure 1), from graph we note that even with 100

2.5 conclusion

the omerta attack with perpetual lead is of diminishing probability of success for security parameter $N \geq 9$.

3 chain fork

fork can occur when multiple stakeholders publish proposal for valid proof of leadership. winning stakeholder can choose to fork with a new block, extend a chain, or extend chanonical longest chain.

3.1 fork finalization

finalization is a compound of longest chain, and single leadership frequency, at the end of any slot, finalization, and re-syncing happen when there is single leader per slot, single leader frequency is controller with secondary discrete PID controller in cascade control. such mechanism is resilient to bribery attack [2], and NAS attack (when the stakeholder write a transaction in one chain, and rewrite the transaction in parallel chain of equivalent stake, and equal fork depth, using relatively small stake) since probability of winning leadership with single leader frequency at small stake at more than one slot during the fork depth is of negligible probability.

3.2 finalization degree of freedom

it's possible for single lead stakeholder at the end of a fork to honor certain fork chain based off off-chain agreement, or if the stake stakeholder leads another block in the fork.

3.3 deterministic finalization

if there are multiple chain fork, execute *rank* function on the longest chains, leadership assigned to the highest rank, and block is finalized.

note that same rank collision is of negligible probability, but in such case, the stakeholder can choose either of which if there is such collision between highest ranks.

3.4 instant finality

the single leader controller aka the secondary controller responsible for fork finalization through single lead per slot achieve 30-38% accuracy on the simulation [7], and up to 50% accuracy on test-net (increases over time; seams the controller learn to adapt pretty well; difference between simulation, and test-net is the distribution of random variables, simulation assumes uniform distribution, which seams to not be the case). instant finality through 100% secondary controller accuracy can be achieved through *L2* leadership election chain [4], meanwhile

Algorithm 1 rank

$pks \leftarrow$ **list of fork competing coin's pks**
$\eta \leftarrow random()$ ▷ output of on-chain VRF
$N \leftarrow len(pks)$
$orders \leftarrow []$
while $N \neq 0$ **do**
    $pk \leftarrow pks[N]$
    $order \leftarrow pk\%(\eta >> 3)$
    $N \leftarrow N - 1$
    $orders.append((pk, order))$
end while
return orders

deterministic finalization(see 3.2) already achieves instant finality, since highest order rank in multiple leaders per block resolves the fork, although it's subject to long rang attack (see 3.4)

3.5 long range attack

offline stakeholder can be malicious if won leadership with highest rank without synchronization, or due to network level corruption by malicious attacker that lead to same case of out of sync chains, if stakeholder came online of sudden and propose different chain with highest ranking leadership proof than canonical one, then transaction can be rewritten by validators discarding established chain, and extending proposed chain.

3.5.1 mitigating long range attack

extending only long chains filters out short chains

forking parallel chain requires 50% of staked DRK and has success rate

$$(f * \sigma)^L$$

L is fork length, σ is relative stake, f is probability of winning lottery owning all the stake, from test-net, and simulation f tend to be around 0.6, to take a conservative intuition, f=1, $\sigma = 0.5$. note that if fork length is 1 then validators choose between highest rank, so lowest possible fork length to rewrite transactions is 2, then at 50% of stake at most conservative case at $f = 1$ probability of such attack is $0.5^2 = 0.25$ assuming malicious attacker always gets highest rank (which is random process), meanwhile a realistic intuition to probability of attack at $f = 0.6$ with 51% of stake at lowest possible fork length is $(0.6 * 0.51)^2 = 0.0936$ assuming malicious attacker always get highest rank, assume number of forks is F, then probability of getting sequence of high ranks is F^L, assume the lowest possible number of forks $F = 2$ then the successful attack at 51% stake, $f = 0.6$, $F = 2$, and $L = 2$ have probability of $\frac{0.0936}{F^L} = 0.0234$.

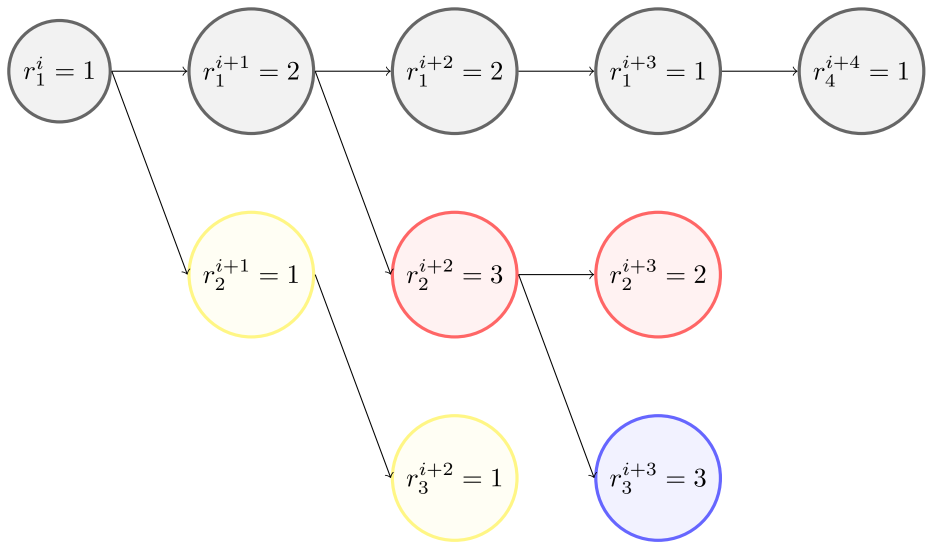

Figure 3: long range attack case study, $r_j^i = k$ means block at slot i of index j has rank k

in span of 4 slots if chain r_1^i,r_2^{i+1},r_3^{i+2} came online at slot $i+3$ (see figure 3.5.1), the attack is possible starting from slot $i+3$, assuming malicious attacker owns 51% of stake, under same previous parameters, probability of attack is $\frac{(\sigma * f)^L}{F^L} = \frac{(0.51*0.6)^2}{3^2} = 0.01$, probability of attack under same parameters owning 10% of stake is 0.00039% in other words the possibility of rewriting transactions loaded in blocks r_1^{i+1}, r_2^{i+2} by malicious attacker that only sync at slot $i+3$ owning block proofs r_2^{i+1},r_3^{i+2} owning 10%, 51% of stake at same parameters above is successful once in 2500, 100 attempts respectively.

4 time frequency

blockchain lifetime is series of epoch

4.1 epoch

multiple slots during which the reward value is fixed, and set by primary controller in the cascade control mechanism [3], and stake is dynamic different from the concept of the epoch in ouroboros, in which the epoch has frozen stake for leadership purposes.

4.2 slot

period of time in which new block is issued

4.3 span

leader election span of time, which can be unspecified period using controller, or fixed 1/3 of slot as described in khonsu [4] during which leaders for slots in the future are elected.

5 timelocked contract

time locked contracts is part of contract, and consensus validation rules, contract locked for n epochs is locked by committing to epoch index used in burn proof constrained coin.

5.1 grinding attack in stake, unstake contracts

stakeholders can move funds between different accounts during the slot for higher probability of winning, such attack can be prevented via timelocked contracts.

6 fee mechanism

slot is clock based, of fixed time length, as a result there is single target of computational cost normal computer (to be defined) can process in a single slot, variable block gas mechanism as in eip1559 won't work.

6.1 fixed slot length

fixed slot length dictates that there is limited amount of computation possible during single slot block gas, or block computational cost is fixed let's call it C, let's distinguish between block computational cost capacity CC, and actual block cost C.

6.2 fee volatility

sudden increase/decrease in demand to fixed supply would lead to spikes in fee cost, and it' can't be accommodated for nor maintained increase in demand, such issues are not auction mechanism design issue, but scalability issues that could mitigated through zk rollups, sharding, parallel tx processing, faster consensus protocols, etc.

6.3 base fee burning

burning base fee b based off parent blocks computational cost, with two conditions: base fee is based off previous blocks computational cost to avoid malicious attacks from current miner to drive fee up, secondly base fee is deterministically calculated rather than agreement between user, and leader in which collusion can occur off-chain to evade base fee burning.

6.3.1 why fee is burned

reduce inflation, raise DRK value. increase cost of spamming, and fake transaction.

6.3.2 tatonment and control

eip1559 [5] propose base fee update rule and discovery of market clearing price, and report on the protocol [6] propose interpolation of base fee between the two different block gas costs at 12.5M, and 25M for fine grained base fee results. however due to fixed slot length, and inability to mitigate sudden increase/decrease in demand more fine grain base fee technique is required, that tatonment addressed above is bad approximation of PID controller.

base fee based off PID control fine grain tatonment can be achieved through PID controller that can reduce spikes in base fee price due to change in demand, with feedback from previous block's cost C. for example base fee f at block i is

$$f[i] = f[i-1] + K_1 e[i] + K_2 e[i-1] + K_3 e[i-2]$$

where $e[i]$ is error function in previous block i-1, K_1, K_2, K_3 are discrete controller parameters (to be tuned). $e[i]$ is the difference between block computation cost capacity, and block cost:

$$e[i] = CC[i-1] - C[i-i]$$

note that block computational cost capacity CC has to be slightly lower than actual computational cost ACC to enable controller to adapt to high demand, for example CC=0.98ACC.

6.4 tip

tip is necessary for transactions inclusion in the block by the block lead.

6.4.1 tip first price auction

tip is similar to eip1559 is a priority fee implemented as first price auction due to simplicity, and undue trusted third party required in more efficient auctions i.e vickery auction. client/user, or tx creators propose tx tip t, assuming client's fee $= t + b =$ max cap fee, then auction utility is nonzero, and max cap isn't required to be added by client in tx header, only tx tip is sufficient.

6.4.2 tipless fee mechanism

for better wallet UX, tip auction can be replaced by tip being function of tx computational cost for inclusion in the block, in such cases, all transactions are priced the same per computational unite, but in cases of high demand, lead, and user can collude off-chain to prioritize transactions, other than that, the collusion can't prevent burning fee, or cheating. note that even if base fee, and tip got low in value due to inflation, it still will not incentivize off-chain collusion, as long as the controller adapts fee to block parent blocks cost C

6.5 computation unites

since slots has limited computation unites, computation unites is better suited for fee estimation than gas price based of block size, or type of contract, every dedicated contract call better have some unite of computation for more accurate estimation of cost, otherwise, lead might ditch computationally expensive txs. in that sense, there are two attributes of txs, tips, and computational cost.

6.6 delay due to empty slots

empty slots with no leads would accumulate pending txs, and would increase base fee on subsequent slots, and users would increase tips for inclusion.

References

[1] Ouroboros Genesis, *Composable Proof-of-Stake Blockchains with Dynamic Availability*, Christian Badertscher, et al, 22-02-2019

[2] bribery attack, *Lecture 18: Bribery and stake grinding attacks*, David Tse, Stanford University, Spring 2020 04-06-2020

[3] cascade, *consensus high incentive monetary policy*, ertosns, 15-04-2023

[4] khonsu, *darkfi khonsu consensus*, ertosns, 31-12-2022

[5] eip1559, *EIP-1559: Fee market change for ETH 1.0 chain*, Vitalik Buterin, et al, 13-04-2019

[6] An Economic Analysis of EIP-1559*, *Transaction Fee Mechanism Design for the Ethereum Blockchain*, tim roughgarden, 01-12-2020

[7] lotterysim, *darkfi lottery simulation*, ertosns, 11-1-2023