Constellation and The Unknowns (Part 2)

Is it Still Possible For a Leader to Frontrun?

Constellation starts with this key point in the abstract

> Constellation ensures that latency and ordering of transactions are fair, fostering a market structure in which users cannot be abused.

So a natural question is if this goal is achieved by current design. Specifically

> Is it still possible for a leader to frontrun transactions? If yes, what are the limits?

In this part 2 of Constellation and The Unknowns we are going to explore this possibility under leader perspective. Part 1 - At Which Batch Does FBO Make Sense? - available here.

Here we will see:

- How at first approximation frontrunning still remains a possibility
- How this highly depends on time when the tx is made available to the leader
- How this type of attack, when considering full path latency, becomes extremely unlikely for tx generated at cycle c where the attacker is the leader
- How this attack still remains a non-negligible possibility, even under Constellation, when the attacker is assumed to operate between consequent cycles
- How in principle this attack can be mitigated acting on the attestations pipeline
- However, it is worth mentioning that Constellation is meant more to solve for censorship, as clearly stated in the abstract, not for all form of MEV that are now possible under leader monopoly.

Transaction Lifecycle

Here we refresh how the lifecycle of a transaction under Constellation looks like. We deliberately stop the process at the leader.

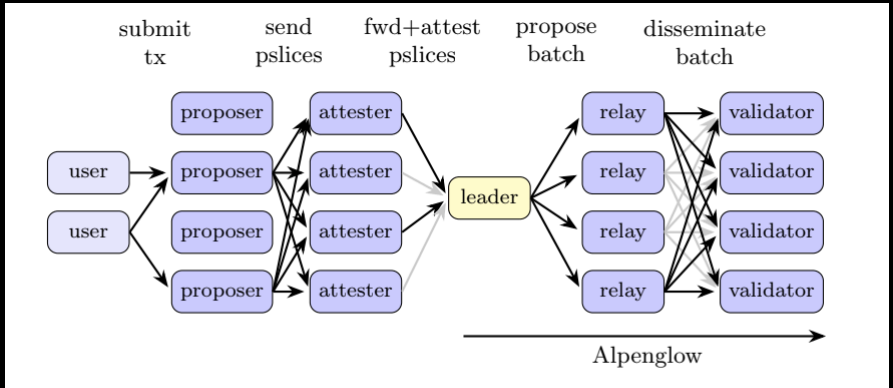


Figure 1: Transaction lifecycle in the Constellation protocol. Gray arrows illustrate redundant packets for resilience, where half of the incoming packets are enough to proceed.

Schematically:

1. User send a tx to one or more proposers
2. During a cycle, each of the p proposer can select transactions, create a pslice and erasure-code them into q pshreds. They send pshreds to attesters. It is worth noting that a proposer can create a maximum of μ pslices per cycle.
3. Attesters immediately send these pshreds to the leader (and few next leaders). Attesters provide also an attestation forwarded at the end of the cycle.
4. Leader compiles all the transactions with enough attestations and produces the batch for each cycle.

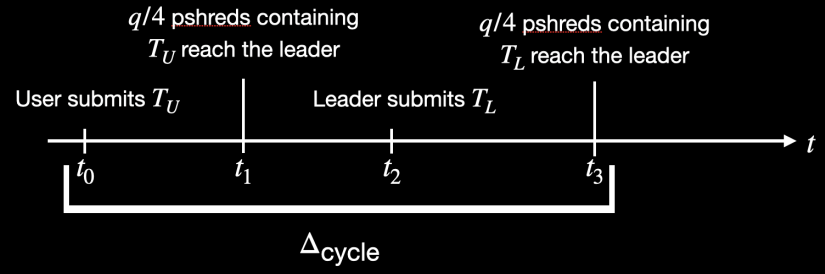
Each pslice is erasure-coded into pshreds with 4x redundancy. It follows that, to recover a pslice you need q/4 pshreds (or q/4 attestations).

The relevant part is that the leader waits q/2 attestations before being able to build the batch, but any tx can be included in a batch if it has q/4 attestations.

Anatomy of the Attack

Here the attack requires 4 events to happen, with a constraint.

1. The user send a tx to proposers
2. q/4 pshreds arrive to the leader from attesters
3. he leader can reconstruct the pslice and get information about the user tx, then can send the frontrun tx
4. the leader tx reach q/4 attesters and receive back relative pshreds



If we denote with t_3 the time at which point 4 is achieved, the leader needs $t_3 \leq \Delta_{\text{cycle}}$, where Δ_{cycle} is the time of the cycle.

Probability of successful frontrun

Let's define with $s_{T_L}(\tau_{T_U})$ as the first available time at which the proposer can include the leader tx into a pslice. The residual time budget before the cycle ends is then

$$B(\tau_{T_U}) = \Delta_{\text{cycle}} - s_{T_L}(\tau_{T_U}) .$$

If $R_i^{T_L}$ is the time needed post proposer to reach the attester of pshred i, we can define the probability the pshred is attested and reach the leader at time t as

$$F_{T_L} = Pr[R_i^{T_L} \leq t] .$$

Now, we can define the Bernulli variable

$$I_i^{T_L}(\tau_{T_U}) = \mathbf{1}\{R_i^{T_L} \leq B(\tau_{T_U})\} ,$$

which is equal to 1 if attester i receives its pshred containing leader's tx in time to attest it in the current cycle, and 0 otherwise.

The number of attesters that receive T_L in time is

$$N_{T_L}(\tau_{T_U}) = \sum_{i=1}^{q} I_i^{T_L}(\tau_{T_U}) .$$

Now, from Definition 9 of the Constellation v0.9 paper, we have that a tuple is attested in A if A includes at least γ attestations. It follows that, the successful event is obtained when

$$\left\{ N_{T_L}(\tau_{T_U}) \geq \gamma \right\} .$$

If we assume independence between attesters, we can define the marginal probability

$$p_i(\tau_{T_U}) = Pr\left[I_i^{T_L}(\tau_{T_U}) = 1 \,|\, \tau_{T_U} \right] ,$$

which depends on attester i: some attesters can be near proposer, some can be far away, some can share the same datacenter etc.

It follows that, under these assumptions

$$N_{T_L}(\tau_{T_U}) \sim \text{PoissonBinomial}(F_{T_L}^1(B(\tau_{T_U})), \ldots, F_{T_L}^q(B(\tau_{T_U}))) .$$

By assuming the probability to reach attesters in time is independent from attester, we get

$$N_{T_L}(\tau_{T_U}) \sim \text{Binomial}(q, F_{T_L}(B(\tau_{T_U})))\,.$$

Hence, the probability of a successful frontrun from the leader is

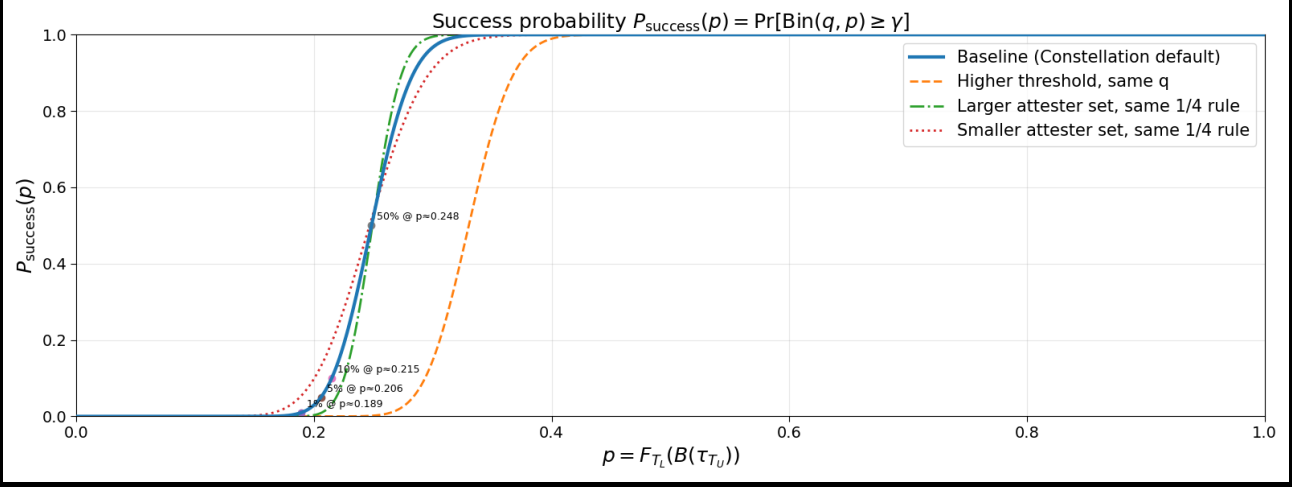
$$P_{\text{success}}(\tau_{T_U}) = Pr\left[N_{T_L}(\tau_{T_U}) \geq \gamma\right] = \sum_{k=\gamma}^{q} \binom{q}{k} \left(F_{T_L}(B(\tau_{T_U}))\right)^k \left(1 - F_{T_L}(B(\tau_{T_U}))\right)^{q-k}\,.$$

This result is particularly useful since we now can study the probability successful frontrun without explicit assumption on the probability to reach γ attesters. That is, if we call

$$p = F_{T_L}(B(\tau_{T_U}))\,,$$

we have

$$P_{\text{success}}(p, \gamma, q) = \sum_{k=\gamma}^{q} \binom{q}{k} p^k \left(1 - p)\right)^{q-k}\,.$$



From this we see that, if the probability the leader's tx reach attesters in time goes above 20%, the frontrun success starts to become a highly non-negligible probability. To have a probability of success below 0.001%, we need to reach attesters in time with a probability lower than 15%.

By changing parameters we see that, to meaningful shift the likelihood of success we need to increase the threshold for eligible attestations, otherwise the result is mostly independent from Constellation parameters.

So, the question is:

> Under what conditions is the probability of reaching the signatories in time low?

To answer this, we can't use a closed form and we need to use a simulation framework.

A MC assessment of the attack probability

Here we develop the MC framework needed to compute the probability of a successful frontrun.

Indeed, the binomial assumption requires homogeneity and independence of attesters. But reality is that attesters have different geolocation, proposers are distributed, and there are common bottleneck.

In this section we assume there is no collaboration between leader and proposer. This basically means that all proposers are not behaving maliciously and are following the pseudocode in the figure below.

Algorithm 1 Proposing for node v, epoch e

```
upon ∃c, j : now() = c · Δ_cycle and proposer(e, c)[j] = v do
    t_0 = (c − 1)μ
    for i ∈ [1, ..., μ] do
        txs ← selectTransactions()                ▷ proposer-specific logic
        if txs ≠ [ ] then
            t ← t_0 + i
            h ← h(txs)                            ▷ see Definition 2
            pshreds ← newPshreds(e, c, j, t, h, txs)
            for k ∈ {1, ..., q} do
                send pshreds[k] to attester(e, c)[k]
        wait until now() = c · Δ_cycle + i · Δ_cycle/μ
```

That is, each pslice is produced in multiples of $\frac{\Delta_{\text{cycle}}}{\mu}$, which otherwise is not enforced by the protocol (as for the current v0.9 version).

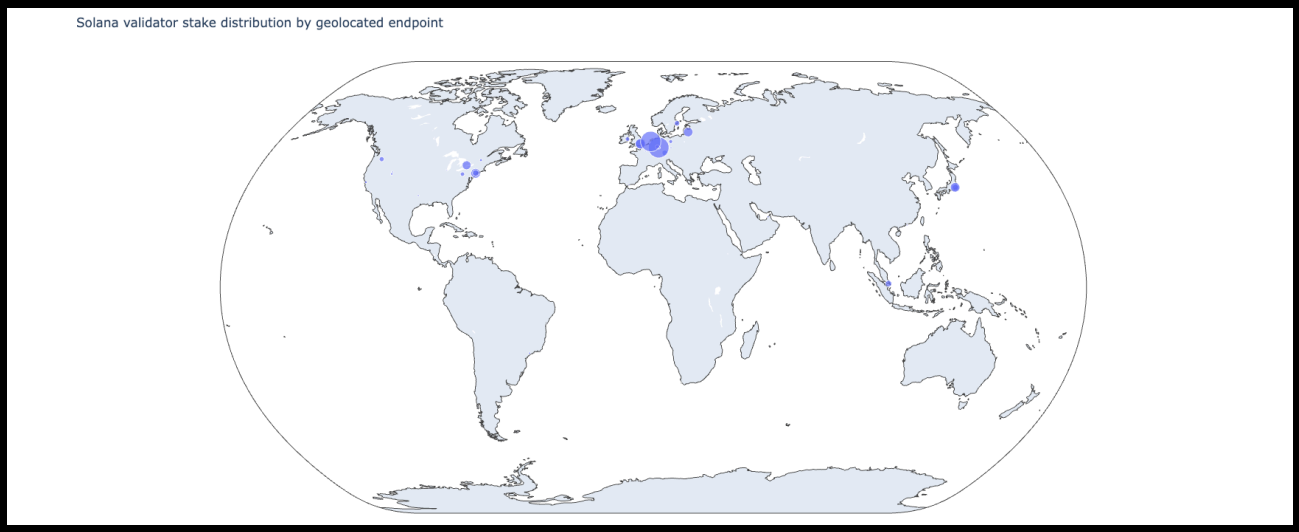
The first available time at which the proposer can include the leader tx into a pslice is

$$s_{T_L}(\tau_{T_U}) = \begin{cases} \frac{\Delta_{\text{cycle}}}{\mu} & 0 < \tau_{T_U} \leq \frac{\Delta_{\text{cycle}}}{\mu} \\ 2\frac{\Delta_{\text{cycle}}}{\mu} & \frac{\Delta_{\text{cycle}}}{\mu} < \tau_{T_U} \leq 2\frac{\Delta_{\text{cycle}}}{\mu} \\ 3\frac{\Delta_{\text{cycle}}}{\mu} & 2\frac{\Delta_{\text{cycle}}}{\mu} < \tau_{T_U} \leq 3\frac{\Delta_{\text{cycle}}}{\mu} \end{cases}$$

where we are considering only the times such that the leader tx is included in the same cycle. It follows that, the residual time budget before the cycle ends is

$$B(\tau_{T_U}) = \begin{cases} \Delta_{\text{cycle}} - \dfrac{\Delta_{\text{cycle}}}{\mu} & 0 < \tau_{T_U} \leq \dfrac{\Delta_{\text{cycle}}}{\mu} \\ \Delta_{\text{cycle}} - 2\dfrac{\Delta_{\text{cycle}}}{\mu} & \dfrac{\Delta_{\text{cycle}}}{\mu} < \tau_{T_U} \leq 2\dfrac{\Delta_{\text{cycle}}}{\mu} \\ \Delta_{\text{cycle}} - 3\dfrac{\Delta_{\text{cycle}}}{\mu} & 2\dfrac{\Delta_{\text{cycle}}}{\mu} < \tau_{T_U} \leq 3\dfrac{\Delta_{\text{cycle}}}{\mu} \end{cases}$$

To get the stake distribution we use a current Solana snapshot.



In our framework, we fix the leader using a node in the network in order of being able to compute the geographical distance between leader-proposers-attesters.

We initially assume the time needed for the leader to receive and reconstruct the user tx as input $\tau_{T_U} \in [0,50]$.

So, we are asking what's the probability of success once the user's tx takes some time to arrive the leader.

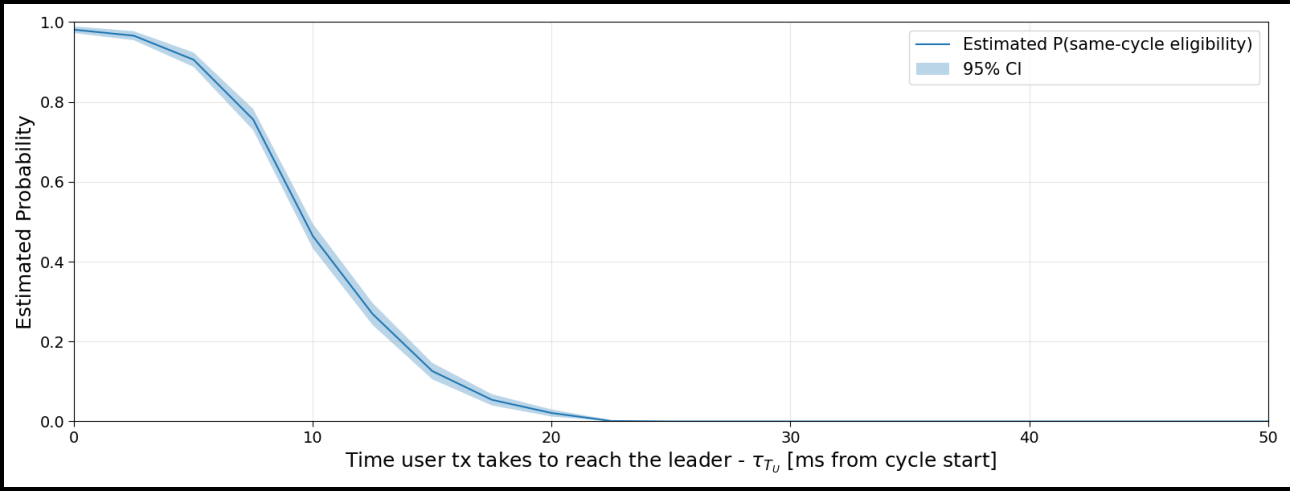
- The latency model parameters of our MC simulations are:
- effective_km_per_ms (eff_ms)= 120.0 → geographical distance requires time for signals to travel
- base_overhead_ms (base_ms)= 2.0 → this adds extra latency even in case of negligible distance (due to eventual bottlenecks)
- multiplicative_sigma (σ)= 0.35 → this controls the multiplicative lognormal jitter
- cycle_common_sigma_ms (cycle_ms)= 1.5 → this models attesters correlation (e.g. in case of congested network).
- country_common_sigma_ms (country_ms)= 1.0 → this models common shocks for nodes in the same country

This means that, latency is modelled as

$$D = (base\_ms + eff\_ms + cycle\_ms + country\_ms) \times \text{LogNormal}(-0.5\sigma^2, \sigma)\,.$$

This model breaks independence between attesters.

Proposers (16) and Attesters (256) are extracted from the stake snapshot in a stake-weighted way. We also allow for the same validator to be attester and proposer since this is allowed by the design.



The malicious validator is fixed to be the node with highest stake in the network, and we assume it's always the leader in our simulations.

At this point, for each simulation, we can compute the time the tx requires from user submission to leader submission of frontrun (and relative time needed to reach attesters). We also account for the residual time budget before the cycle ends.

We consider the leader sending his tx to all proposers.

Each simulation produces an indicator of successful event (i.e. leader tx reach enough attesters) $S^{(m)} \in \{0,1\}$. Repeating for all simulations, we can define the Monte Carlo estimator of probability of same-cycle eligibility

$$\hat{P}_{\text{success}} = \frac{1}{N_{\text{sims}}} \sum_{m=1}^{N_{\text{sims}}} S^{(m)}.$$

The confidence interval is then computed as standard binomial error

$$\sqrt{\frac{\hat{P}_{\text{success}}(1-\hat{P}_{\text{success}})}{N_{\text{sims}}}}.$$

We can see that, if the tx arrives to the leader after 20ms the probability of same-cycle eligibility is basically zero.

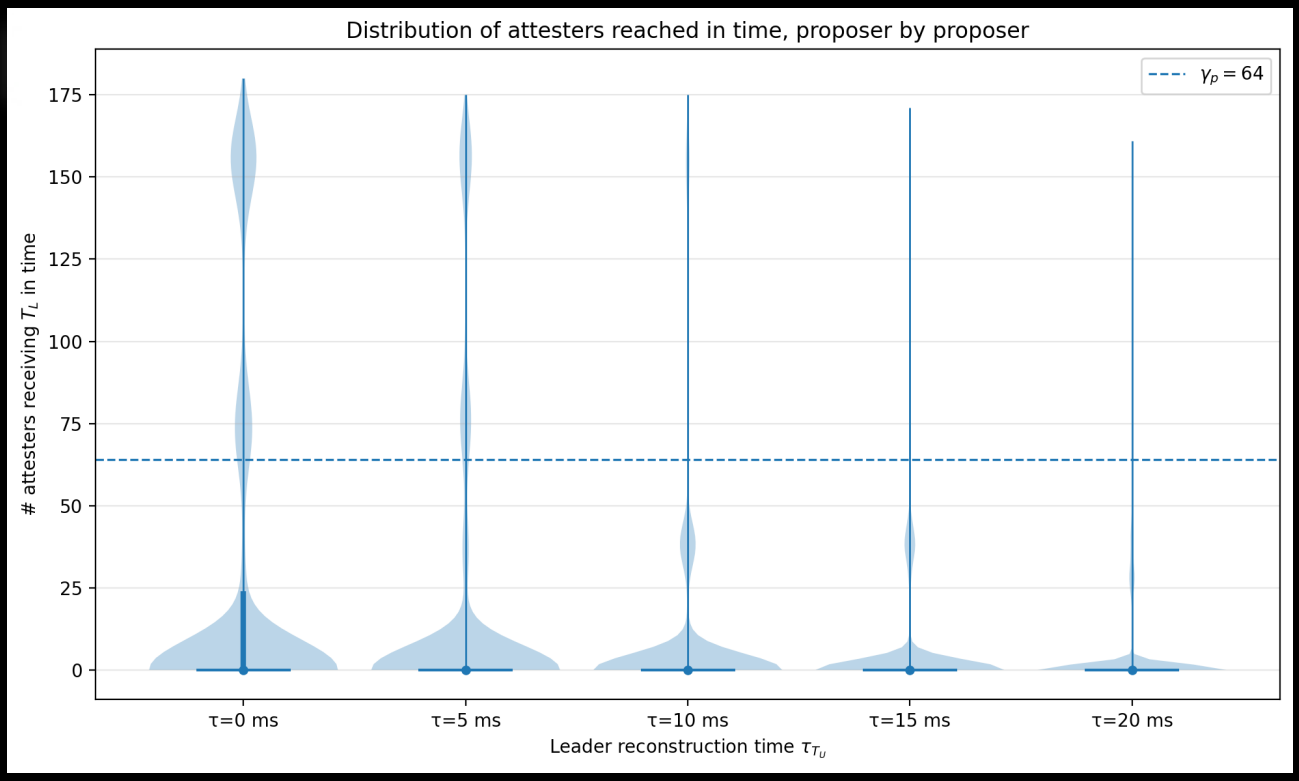
If we check for the distribution of attesters reached in time, proposer by proposer, we see that this is an effect due to the fact that only few proposers really reach all needed attesters.

This means that this strategy may works only if the leader send the tx to all proposers, actually diluting the profit (due to inclusion fee payments).

Full Path to The Leader

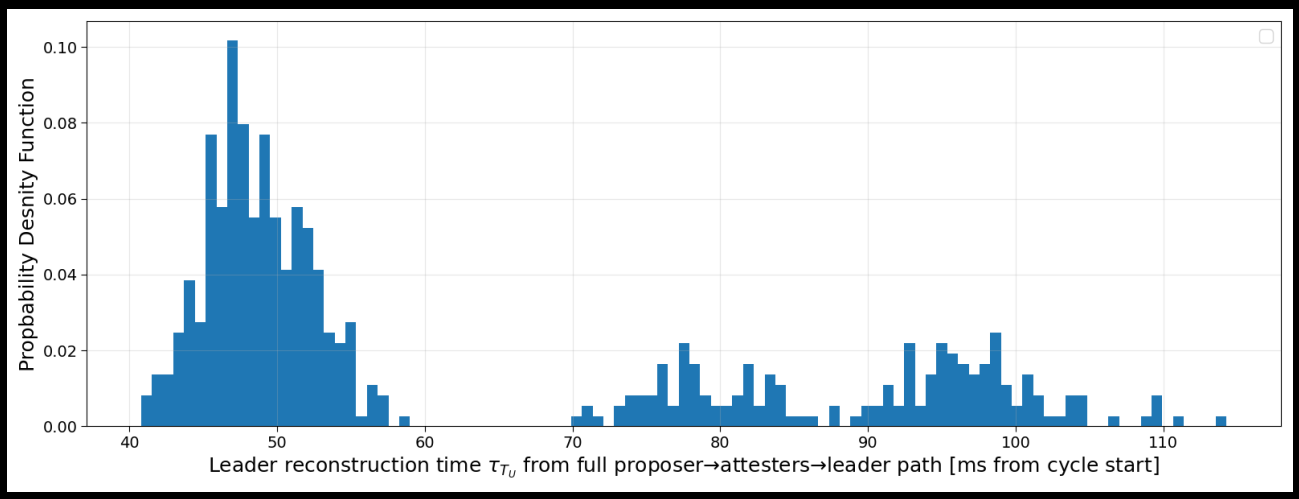
Ok, now we have seen how, provided that the tx arrives before 20ms to the leader, the leader can perform a frontrun attack with a non-negligible probability of success.

However, one question stands:



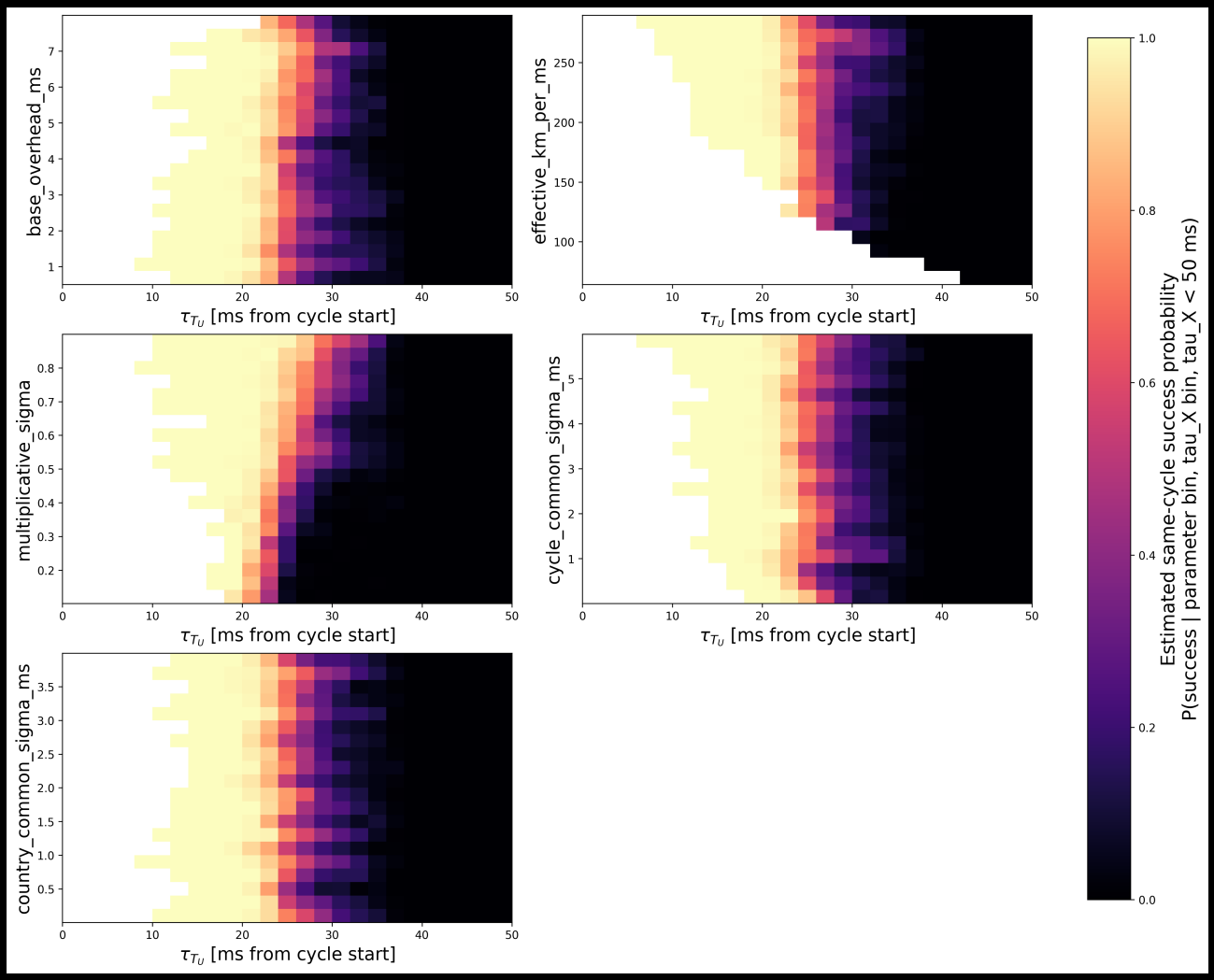
Under our assumptions, how likely it is a tx arrive to the leader before 20ms from the start of the cycle?

To answer that, we employ the same framework and simulate for the full path proposers → attesters → leader.



The simulated distribution of leader reconstruction times shows that, under the honest-proposer grid model, same-cycle insertion of leader's tx is effectively ruled out, since user's tx is never reconstructed by the malicious leader before the last proposer send opportunity at 37.5ms (with Constellations parameters).

However, a non-zero fraction of transactions are reconstructed only after the end of the current cycle, which opens a potential cross-cycle (and even multi-cycle) attack surface.



Sensitivity Analysis

As any model, most of the results from these simulations are dependents on the parameters. This clearly introduce some sort of possible bias due to what are the prior assumptions of who develop the analysis.

For this reason, one of the way to remove the bias and explore all possibilities is to start varying all parameters in the latency model assumption.

In this section we ask what is the probability of a successful frontrun given a specific set of parameters, when we ask that the user tx takes less than 50ms to reach the leader.

The joint parameter sweep shows that high same-cycle success probability is concentrated in network regimes that combine low fixed hop latency, high effective propagation speed, and either substantial path heterogeneity or correlated favourable shocks.

Here, base_overhead_ms captures fixed non-geographic delay such as serialization, queuing, and software/network-stack overhead, so lower values directly favour the attack.

By contrast, effective_km_per_ms controls how strongly geographic distance is penalized; high values correspond to near-fiber, and in the extreme near-light-speed, end-to-end propagation, so the regions of highest success at very large values of this parameter should be interpreted cautiously as physically optimistic.

The parameter multiplicative_sigma does not make the network uniformly faster, but increases dispersion in proposer-to-attester latencies; this thickens the lower tail and therefore increases the chance that at least one proposer attains unusually fast paths to a subset of attesters large enough to cross the eligibility threshold.

Similarly, cycle_common_sigma_ms and country_common_sigma_ms capture cycle-wide and geographically clustered co-movements in latency; large values do not reduce the mean delay, but create favourable realizations in which many related paths become simultaneously fast.

Overall, we can see that the same cycle probability is non-negligible when we are in a favourable environment for the attacker. High effective propagation speed is helpful, but not sufficient on its own; the attack is strongest when it is combined with enough path heterogeneity for at least one proposer to reach a threshold-sized subset of attesters unusually quickly. Thus, Constellation does not rule out same-cycle frontrunning, but confines it to favourable timing and network conditions.