Committee Selection Risks in Midnight

Version 1.2.2 | 2025-03-24

Jon Rossie

This document is actively maintained. See **Status & Changes**, Section A, for recent updates and roadmap.

Table of Contents

Contents

1 Overview **3**

2 Finality Risk **4**
2.1 Quorum-Based Consensus and Weighted Selection 4
2.2 Committee Selection 5
2.3 Voting Strength 6
2.4 Implications for Risk 6
2.5 Subsetting of Candidates 7
2.6 Quorum Assumptions and Pathologies 8
2.7 Protocol Specification and Implementation Risks 10

3 Objectives and Measures **10**
3.1 Objective: Liveness in the Face of Crash Faults 10
3.2 Primary Risk Measure: Number of Faults Tolerated 11

4 Inherent Risk **12**
4.1 Approach 12
4.2 Limitations 14
4.3 Analysis 14

5 Ariadne for Risk Prevention **14**

6 Ariadne Residual Risk **18**
6.1 Simulation Parameters and Approach 18
6.2 Simulation Results 18
6.3 Analysis 23
6.4 Recommendations 23

7 Toward a Full Model of Finality Risk **23**
7.1 The Risk of a Committee that Cannot Tolerate f Faults . . . 25
7.2 The Risk of f Faults Occurring at Once 25
7.3 Putting it Together 28
7.4 Telling Risks Apart 28
7.5 Conclusion 29

A Status & Changes **30**
A.1 Version History 30
A.2 Future Roadmap 30

B Risk Management Framework **31**
B.1 Risk Decision-Makers & Appetite 31
B.2 Impact vs. Probability (Severity vs. Frequency) 32
B.3 Risk Event Types & Categories 32
B.4 Inherent Risk (Pre-Control Assessment) 32
B.5 Risk Controls & Residual Risk 33
B.6 Monitoring & Adaptive Risk Management 33

C Unused Charts **33**

1 Overview

This document is concerned with a specific risk Midnight faces: the risk of selecting committees with low fault tolerance, thereby increasing the risk of a breakdown of the consensus protocol, GRANDPA, that finalizes blocks. Finality plays a critical role in Midnight because, unlike most blockchain clients, Midnight's smart-contract client `midnight.js` is unable to process rollbacks. When the client observes a confirmed transaction, it applies pending local state updates in a destructive manner. Because of this (perhaps temporary) limitation, Midnight has been designed to assume fast finalization to reduce user-visible latencies while protecting against data loss. Fast finality also provides direct usability benefits for people interacting with the chain, so Midnight's commitment to fast finality is never likely to decrease.

Figure 1 shows a simplified set of Midnight components. The Midnight Node observes its Cardano Node via DBSync, and does so at a delay of 12 hours because that is Cardano's longest-chain finality window. But most Midnight clients are more concerned with the finality delay for transactions they post to the Midnight Node. Midnight clients rely on the Indexer to provide only finalized Midnight transactions, to prevent possible data loss should rollback occur on the Midnight Node. Anything that disrupts the orderly finalization of blocks will impact clients. If GRANDPA experiences liveness issues that delay finalization for unbounded periods, Midnight will appear offline to its clients even if new blocks are being produced. If GRANDPA experiences a safety violation that allows it to finalize contradictory forks, clients will struggle to recover.

Although this document mentions GRANDPA, none of the analysis is specific to that protocol. The entire analysis is relevant to any consensus model that uses quorum-based BFT with stake-weighted, decentralized committee selection. This includes the potential replacement for GRANDPA and AURA being prototyped as part of the FastBFT effort, based on Jolteon consensus.

The rest of this document presents a detailed analysis of Midnight's finality risks arising from stake-weighted committee selection. After analyzing the inherent risks of that approach, we describe a variant, Ariadne, developed for Midnight. We analyze its effectiveness in risk reduction and offer recommendations for dynamic tuning of Ariadne's parameters for best effect. We conclude with an overview of how this document's analyses fit within a larger risk model of quorum-based voting in Midnight.

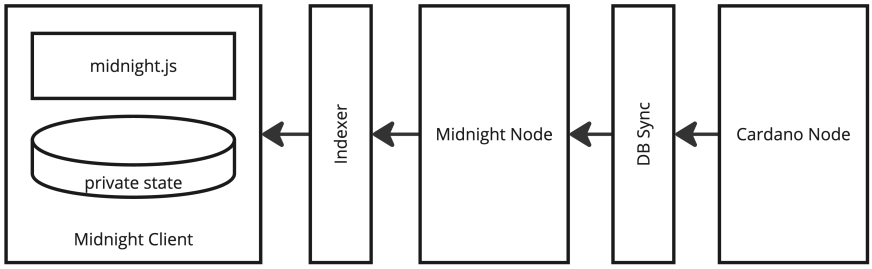

Figure 1: *Major components of the Midnight architecture.*

2 Finality Risk

To understand finality risk, we must understand how the consensus protocol makes progress and provides safety. We will then review the ways in which the protocol can fail, and the effects of those failures.

2.1 Quorum-Based Consensus and Weighted Selection

Midnight uses a proof-of-stake consensus where the amount of control wielded by a community member is proportional to their stake relative to the stake of other participants. Unlike Cardano's Ouroboros proof of stake, which is a longest-chain model, Midnight uses quorum-based consensus to achieve consensus with faster finality.

Quorum-based consensus in the presence of adversarial actors is a well-studied problem, often termed *classical Byzantine Fault Tolerance* (BFT). A *Byzantine fault* encompasses any fault presenting different symptoms to different observers, including both malicious behaviors and inadvertent faults or network conditions causing a well-intentioned participant to deviate from correct protocol behavior.

A fundamental result in this area is that quorum-based BFT consensus requires at least $3f + 1$ participants to tolerate f concurrent Byzantine faults. In contrast, longest-chain protocols such as Ouroboros operate under a weaker assumption, necessitating only $2f + 1$ participants (commonly referred to as a 51% honest majority). However, this comes at the expense of longer finality times. Quorum-based voting can achieve finality in seconds to minutes, whereas longest-chain protocols may require hours to confirm

transaction immutability.

Throughout this discussion, we distinguish between crash faults and Byzantine faults. While crash faults represent a subset of Byzantine faults from the perspective of the overall network—since other participants typically cannot distinguish between a crashed node and a Byzantine node—the distinction is significant for node operators. Operators can quickly detect and respond to crash faults using simple process monitoring. In contrast, Byzantine faults may be subtle or concealed, causing the operator to remain unaware of their node's faulty behavior and thus delaying diagnosis and recovery.

2.2 Committee Selection

Quorum-based consensus is built on the notion of a *committee*—or series of committees—whose members propose and vote on the value for each new slot. (In our case, they propose and vote on the next block in the chain.) In a decentralized system, this means there must exist a *committee-selection* algorithm that chooses a committee from a set of candidate participants. Midnight follows an approach where committee selection is based on *weighted random selection* using each registered participant's stake as their weight. The rough outline of the procedure is:

- Let $P = \{(p_1, s_1), ..., (p_n, s_n)\}$ be the set of candidate participants p and their stakes s at the time of committee selection. Assume P is a subset of some larger set A of all stake pools.
- Compute $S = \sum_{i=1}^{n} s_i$ as the total stake.
- Compute each participant's weight $(p_i, w_i) = \frac{s_i}{S} \quad \forall (p_i, s_i) \in P$. Let W be the set of all such weights.
- To fill a committee of size k, make k separate weighted selections from W, allowing the same participant to be selected many times. The resulting committee is an ordered list $K = [p_1, ..., p_k]$ which can include duplicate entries. We may refer to p_i as the *owner* of the i^{th} committee seat.

We assume the overall system selects new committees at some configurable interval, running this full algorithm each time.

2.3 Voting Strength

Participants selected into a committee do not all wield equivalent voting power in consensus. This is an important difference from a typical centralized deployment, where every node's intentions are trusted equally and every node's exposure to unintentional faults is roughly equivalent. Variable voting strengths are a deliberate outcome of stake-weighted selection. They reflect a complex set of assumptions that can be reduced to "*more stake implies greater trust.*"

The amount of power a committee member holds, called is *voting strength*, is the percentage of total seats it owns on the committee. To compute the voting strength of each participant p in a committee k, first count the number of seats p owns in K; this is expressed as $\text{count}(p, K)$ or C_K^p for short. The voting strength V_K^p of a participant p in a committee K with size k is computed as

$$V_K^p = \frac{C_K^p}{k}$$

Naturally, a candidate who is not selected into a committee has zero voting strength on that committee.

2.4 Implications for Risk

Voting strengths play a central role in the computational model of risk this document develops. For a simple preview: suppose a single participant p has voting strength $V_K^p = 0.34$ on a given committee K. If p were to become faulty, the committee would not be able to achieve consensus on any finality votes, and the system may exhibit either safety or liveness issues (or both), as discussed in section 2.6.

In our fault analysis, we are concerned primarily with the ranked voting strength of committee members. For a committee K with distinct members $\{p_1, ..., p_m\}$, we analyze the voting-strengths vector

$$V_K = [V_K^{p_1}, ..., V_K^{p_m}] \qquad \text{sorted in descending order}$$

This vector allows us to analyze the amount of voting strength at risk under different fault assumptions.

2.5 Subsetting of Candidates

Two critical steps on the path to forming a committee act as filters, reducing the set of active participants. Starting with the full set of potential candidates (say, Cardano SPOs in the case of Midnight), the first filter is whether or not these potential candidates register to be considered for committees; we call this self-selected subset the *participant pool*. Then weighted selection acts as another filter, almost certainly subsetting the participant pool due to the low probability of selecting candidates with very low relative stake for any committee seats. Only those candidates who win committee seats are active participants in consensus.[1]

The subsetting performed by weighted random selection means some candidate participants will be excluded from some committees. This is true even if the committee size k is much larger than the number n of candidates. Whenever a candidate with large stake wins an additional seat, that seat is no longer available to other candidates. The effect when stakes vary widely is to probabilistically exclude the lowest-staked candidates from committees.

If we equate trust with amount staked, this probabilistic subsetting effect increases the security of the chain. But it comes at the cost of potentially centralizing authority among the largest players. This centralization can be reduced by imposing a non-linearity to voting strength to create diminishing returns past a certain stakepool size. Midnight has no direct mechanism like this in its launch plan because Midnight participants are a self-selected subset of Cardano stakepools, and Cardano already has a *saturation* mechanism to implement diminishing returns for extremely large stakepools.

Any mechanism that artificially inflates the voting power of lower-staked candidates opens the door to Sybil attacks in which a large stakeholder transfers their stake to a large number of smaller stakepools it controls, resulting in a much higher aggregate voting strength than their single stakepool would have commanded. This risk justifies Midnight's conservative use of strictly linear scaling when computing voting strength. While the linear approach is not a defense against all forms of Sybil attacks, it avoids introducing a direct incentive among rational actors to split their stake.

[1] The committee members are also the active participants for block production on Midnight, but the focus of this paper is their participation in finality voting.

2.6 Quorum Assumptions and Pathologies

For the purposes of this discussion, we will write $QBFT$ to refer to the broad class of quorum-based BFT consensus protocols, to distinguish them from longest-chain BFT protocols such as Ouroboros. Further, we will refer to $QBFT$ as if it were a single protocol, but our scope includes many practical quorum-based protocols; the properties we discuss here are common to such protocols.

QBFT consensus assumes greater than $2/3$ of the committee are *honest* in the strictly technical sense of *operating within the specification of the protocol*. Committee members with the best intentions may fail to participate honestly if, for example, they experience faults (crash faults or otherwise), or if their network connectivity suffers due to benign or malicious causes.

QBFT does nothing to ensure greater than $2/3$ honest participation or to cope with situations where $2/3$ or less of the committee is behaving honestly. Those responsibilities fall to the system in which QBFT is a component. From the point of view of the QBFT protocol designer, as soon as the security assumption is violated, "all bets are off" and protocol behavior is wholly unspecified. In reality, as we explore below, QBFT systems degrade in some very specific ways depending on how the security assumption is violated. Midnight, as a system leveraging QBFT, must bring its own methods of fostering greater than $2/3$ honest participation, and must detect and recover when this security assumption fails.

2.6.1 What Can Go Wrong?

When a QBFT system's honest-participation assumption is violated, its behavior is unspecified. The ill effects of this unspecified behavior are typically classified as either liveness or safety violations, as described below.

Liveness Risk Liveness risk is the risk that the system will enter a state where progress is impossible. In a strict technical sense, this usually refers to a terminal state (or cycle of states) from which there is no recovery. In our discussion we are also concerned with transient, intermittent states in which no finality votes are possible until either (a) faulty committee members return to normal behavior, or (b) a new committee is formed with sufficient honest members to restore progress.

A QBFT that experiences $f \geq 1/3$ crash faults is below quorum and can no longer agree to finalize new blocks. Depending on the details of the block-production rules, either the system will stop extending the chain with new blocks or it will simply stop agreeing that any new blocks are finalized. In the time it takes the faulty nodes to detect their faults and return to operation, progress on block finalization may be delayed arbitrarily. Also, there is no inherent mechanism to exclude faulty nodes from each new committee, so repeated committee selection is likely to keep including faulty members, prolonging the system liveness fault.

If crash faults alone are problematic, Byzantine faults can be worse. The operator of a node experiencing an unintentional Byzantine fault will have a harder time detecting the fault than they would a crash fault, delaying restoration of normal behavior. If, instead, adversarial operators deliberately withhold votes, they may block (or help block) progress for as long as they are assigned sufficient voting strength in new committees. For example, a single committee member with 34% voting strength can block all progress by ceasing to submit votes.

These potentially unbounded interruptions of the normal process of finalizing blocks are considered liveness faults in our analysis.

Safety Risk A QBFT protocol is *safe* if no two honest participants or observers can witness the protocol making contradictory decisions. In the case of a blockchain, let b_i^F be a block at height i that any witness can observe as having been finalized by the protocol and let $b \prec b'$ signify that b is in the prefix chain of b'. The protocol is *safe* if for all finalized blocks b_i^F, b_j^F, and b_k^F where $i < j < k$, it holds that $b_i^F \prec b_j^F \prec b_k^F$. Simply: the protocol is safe if it does not finalize conflicting forks.

A QBFT system experiencing $f \geq 1/3$ Byzantine faults might fail safety if some of the faulty nodes *equivocate*, which means voting to support conflicting proposals. This is the classic form of Byzantine fault, where observers are somehow partitioned such that they are unable to compare notes and discover that a duplicitous voter is voting differently on both sides of the partition. This could lead observers in the two partitions to witness quorums and support conflicting finality decisions. While *Byzantine* faults refer to any non-crash faults, equivocation is the key safety risk and is therefore the primary focus of Byzantine fault tolerance.

2.7 Protocol Specification and Implementation Risks

The remainder of this document makes two critical assumptions, neither of which can be verified. First, it assumes the GRANDPA protocol's specification describes a system than can maintain safety and liveness as long as there is a 2/3 honest majority. Second, it assumes that the GRANDPA implementation does not introduce defects that cause safety or liveness failures even when a 2/3 honest majority are present. In fact, GRANDPA is not a verified protocol, so its actual ability to provide safety and liveness within the assumed security tolerances is unknown. Moreover, the GRANDPA implementation deviates from the specification in ways that have defeated efforts to establish its actual safety or liveness guarantees. Thus, every assertion in the rest of this document comes with a caveat that the true situation might be worse than we describe. For this reason, the Midnight team are actively supporting the development of a fully formalized replacement consensus system based on the Jolteon BFT protocol.

3 Objectives and Measures

Before we manage a risk, it is useful to quantify it. In this section we identify the most urgent risk objective and our approach to measuring it. The same measurement approach should apply to the system both before and after the introduction of preventive or mitigating controls. In later sections we will analyze the inherent risk (before controls) and residual risk (with controls).

3.1 Objective: Liveness in the Face of Crash Faults

In a system that could easily face deliberate attacks against liveness and safety, there is significant value in focusing on a much simpler path to failure: the risk of crash faults that block new finality votes.

It may seem counterintuitive to focus on these benign scenarios, but it helps to consider that no system that can easily fail for accidental causes is safe from deliberate attacks. Surviving ordinary faults is the baseline for all further risk management.

Put differently, a system that requires the simultaneous failure of f benign nodes has also made it impossible for an adversary to realize an attack without controlling more than f committee seats, either directly or through corruption. Anything we do to drive f higher to protect against liveness

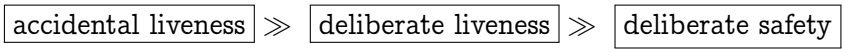

Figure 2: *A risk hierarchy. If we cannot defend against accidental liveness faults, we have no defense against deliberate liveness attacks. If we cannot defend against deliberate liveness attacks, we have no defense against deliberate safety attacks. That is why we focus first on accidental liveness faults.*

failures among well-intentioned committees also reduces the chance of a successful attack on the liveness or safety of the system. Figure 2 illustrates this relationship.

3.2 Primary Risk Measure: Number of Faults Tolerated

Our central risk measure is the probability of selecting a committee in which crash faults of fewer than f nodes can result in a liveness failure. The target f is an organizational decision, part of the organization's *risk appetite*.

Since committees are chosen randomly from participants, which are a random subset of all SPOs, our aim is to predict the probability of a random committee arising that cannot survive a target minimum number of faults. Put differently, we will analyze the highest number of faults that a potential committee can be expected to survive with a target probability. This lets us pick a risk tolerance of, say 99%, and aim for the highest f that can be achieved with 99% probability.

3.2.1 Computing the Number of Faults Tolerated

An intuitive way to understand the number of faults a given committee $K = [p_1, ..., p_k]$ can handle is to first derive $V_K = [V_K^{p_1}, ..., V_K^{p_m}]$, which are the voting strengths of each committee member, sorted from highest to lowest. We can then compute f by running the algorithm shown in Listing 1. The algorithm finds the longest prefix of V_K whose total voting strength is less than $1/3$ the total voting strength of the committee. The length of that prefix is the number of faults the committee can tolerate. Figure 3 illustrates this process in two scenarios with different degrees of fault tolerance.


```
x, f = 0
for v in V:
  if x + v >= 1/3:
     break
  x += v
  f += 1
return f
```

Listing 1: *Counting Faults: This algorithm processes voting-strength vector* $V_K = [V_K^{p_1}, ..., V_K^{p_m}]$ *for a committe* K*, sorted in descending order. The value returned is greatest number of faults that the system is guaranteed to survive. The committee may survive more faults if they occur in lower-strength committee members, but* $f+1$ *faults are capable of violating the security assumptions of the system.*

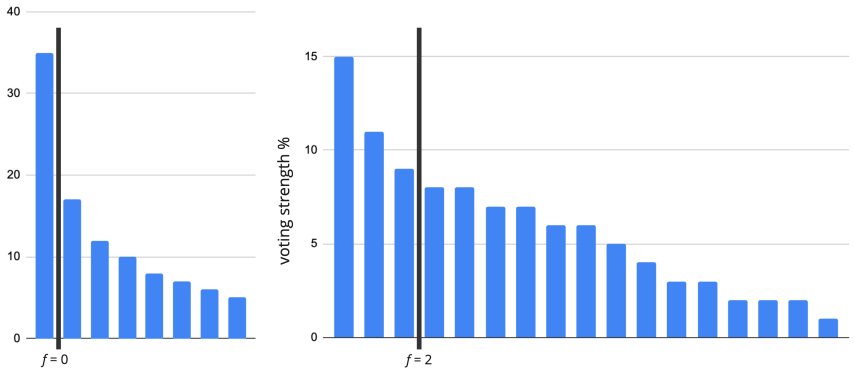

Figure 3: *Counting Faults: This figure illustates the process for counting tolerated faults. Every blue bar is a voting strength in* V_K*, the vertical divider is the final index of the loop, and the value for* f *is the count from the prior step. Read this as "if all the committee members left of the mark fail, the system has failed." As long as only* f *faults occur, the system still has sufficient voting power to finalize blocks.*

Key Concept: What f Means Looking at Figure 3, one might question why we say scenario 1 cannot survive a single fault when it could clearly survive a fault in its weakest committee member. In fact, if we simply sorted the members in increasing order, our loop to discover f would return much higher values for f. Why do we sort in descending order when determining the value of f? Think of f as a number of blank termination notices you give to an adversary. The adversary can halt any f nodes they like. Your job is to pick the highest possible f such that your adversary is unable to put the system in a faulty state regardless of which members they halt. That is the point of f: you always know you can lose *any* f nodes and maintain operation.

4 Inherent Risk

4.1 Approach

To analyze inherent risk we perform a Monte Carlo simulation of the process of selecting random committees with different parameters. The simulation details are:

1. Our SPO data is the set of all SPOs on Cardano at the time of a scrape of `pooltool.io` at some point in September 2024.
2. We then simulate committee selection assuming different numbers of registered Midnight participants. We use a range from fifty up to two thousand, out of roughly 3000 total SPOs. Committee size seems to have fairly little impact on fault tolerance, so we just assume committee size is 300.
3. For each committee generated, we compute the number of faults the committee can tolerate.

We then aggregate data from thousands of such simulation runs to build a probabilistic model indicating the risk that a committee drawn from a registered participant pool of each size will result in a committee with low fault tolerance. This provides the baseline data shown in Figure 4. In this chart, 1.0 means 100% of the simulations tolerated the given number of faults.

4.2 Limitations

Although our simulations are based on real data from `pooltool.io`, we make a naive assumption that *every SPO is equally likely to participate in the Midnight network.* This is surely untrue, but we currently lack data to justify an alternative model. Therefore, we use a flat distribution in our Monte Carlo simulations. We can also construct interactive models to allow experiments where SPOs are grouped (e.g., into deciles), and allow users to specify participation assumptions for each group. While these assumptions cannot be validated in advance, they may still prove valuable for scenario-based risk analysis. The most accurate results will become available once Midnight is live on Cardano, allowing us to employ iterative Bayesian analysis to estimate the probability of various distributions, conditional on the SPOs already participating in the Midnight network.

4.3 Analysis

It's clear from the chart that more participants lead to higher fault tolerance, even with a fixed committee size of 300. At the same time, it's important to note that a centralized consensus protocol could tolerate 21 faults with 124 nodes ($3f + 1$), whereas our decentralized, stake-weighted participation approach requires more than 1000 to reach the same degree of fault tolerance (closer to $300f$).

If Midnight could launch with 1000 active participants, the risk analysis would focus on making their committees survive even more faults. But in the early stages of Midnight it's more realistic to expect 200 or fewer participants. We should be looking for early-stage alternatives or augmentations to simple stake-weighted committee selection that yield committees whose fault tolerance is similar to that of a pure stake-weighted model with 1000 or more participants.

5 Ariadne for Risk Prevention

The inherent risk analysis shows that adding more participants increases the fault tolerance of the system. Ariadne[2] is a modification to committee

[2]Unpublished, IOG internal. Ariadne was conceived as a stepping stone to true multi-resource consensus in Minotaur, designed by Zajkowski and Rossie circa 2022 for Midnight. In classical mythology, Ariadne gave Theseus the ball of string he used to escape the

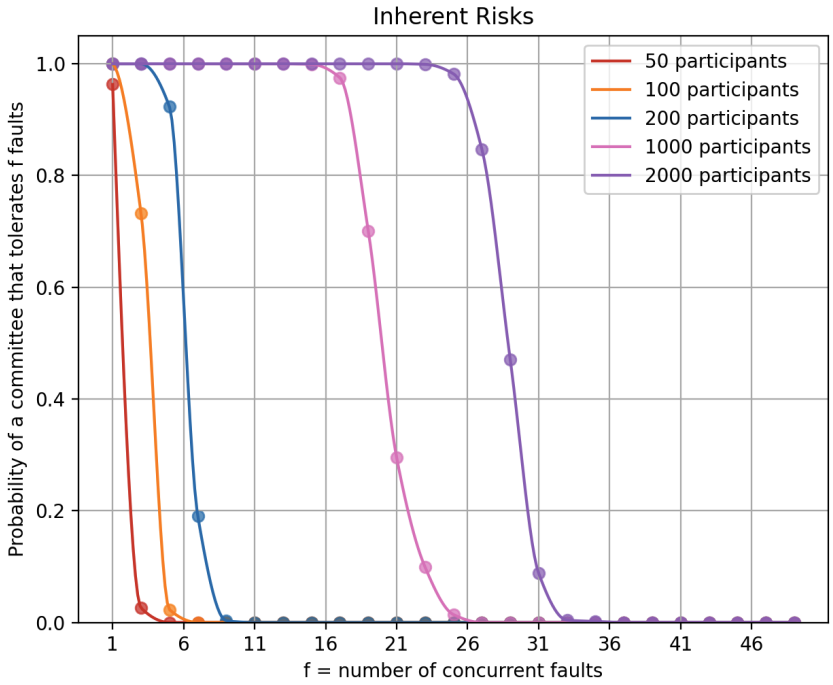

Figure 4: *Inherent risk that a random committee will not tolerate* f *concurrent faults, for* f *in 1-50. The different curves represent different numbers of registered participants. (We acknowledge the labeling here may be confusing. The "risk" is higher at higher values of* f*, as the chance of tolerating that many faults decreases. This chart matches the residual risk charts in later sections, but we may update all these charts and wording in a later revision.)*

selection that adds a relatively small number of permissioned nodes holding sufficient stake to stand in for the eventual SPO participants until they arise.

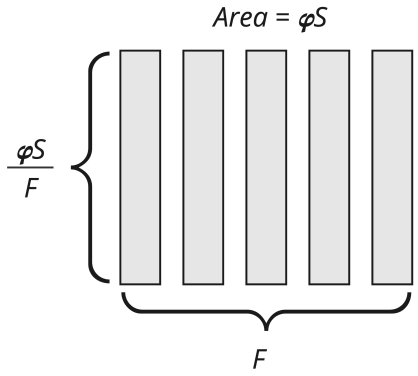

Figure 5: *Ariadne's two key parameters are* φ*, the percentage of voting strength reserved for the federated nodes, and* F*, the number of federated nodes across which Ariadne distributes this voting strength. If* S *is the sum of the stake held by all SPO participants,* φS *is the virtual stake held by federated nodes on the committee. The voting strength of each federated member is* $\frac{\varphi S}{F}$*.*

Figure 6 shows a committee in which Ariadne has reserved five committee seats for its auxiliary *federated* members, and has assigned each an equal amount of *virtual stake*. The effect is to dilute the voting strength of the more strengthful permissionless members, raising the overall fault tolerance.

Figure 5 illustrates the two key parameters that control Ariadne. The first is F, the number of committee seats reserved for federated members. The second is φ which represents the percentage of total voting strength to be held across those federated members.[3] Ariadne divides the federated voting strength evenly across the F federated members.

Ariadne follows these steps to create a committee of size k:

1. Reserve φk committee seats for the federated nodes. (These reserved seats may be randomly distributed across the k committee seats.)

Minotaur's labyrinth.

[3]The actual configuration parameter that defines φ is the "D parameter", which is specified as a ratio of permissioned to permissionless seats.

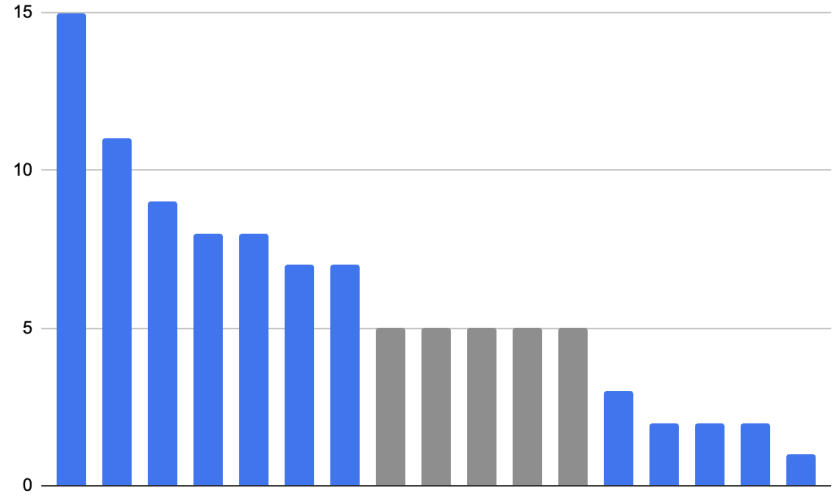

Figure 6: *Ariadne-inserted committee members dilute the voting strength of the highest-strength members in order to raise the overall fault tolerance of the commitee.*

2. Use weighted random selection to populate the unreserved permissionless $(1-\varphi)k$ seats in the usual way, choosing from the available participants and allowing the same participant to occupy multiple seats.

3. Populate the reserved federated seats in round robin from the set of F federated nodes, again allowing the same participant to occupy multiple seats.

This will result in a committee where each federated node has a voting strength of approximately $\frac{\varphi k}{F}$, regardless of the voting strengths of the permissionless nodes. When the voting strengths are sorted, the federated nodes will sort together in a block as illustrated in Figure 6.

The federated nodes must be explicitly configured in Midnight. They are sometimes called *trusted* nodes because they are expected to behave honestly within their ability. But as with any nodes, federated nodes are subject to their own faults. The fault analysis of an Ariadne committee therefore follows the same approach used in our study of inherent risk, but with federated nodes inserted in the correct sorted position within the list

of committee members, as illustrated in Figure 6.

6 Ariadne Residual Risk

To measure the risk reduction offered by Ariadne, we use Monte Carlo simulations to show the effects of Ariadne with changing parameters.

6.1 Simulation Parameters and Approach

The key variables we test are:

1. The number of participating SPOs, drawn from the Sept 2024 pooltool scrape.
2. The φ parameter—the percentage of voting strength reserved for the federated nodes.

We keep two important parameters fixed:

1. We set the number of federated nodes at 10. That number is $3f+1$ for $f = 3$, allowing the federated nodes to act as their own fault-tolerant cluster for ancillary consensus problems that can support Midnight in other ways. Ten is a reasonable number in terms of operational cost and complexity. More is always better from a risk perspective, but practical concerns suggest a number closer to 10.
2. We fix the committee size at 300, as it has little effect on the outcomes (as discovered in prior experimentation).

6.2 Simulation Results

Here we present our results as a series of charts, each showing the fault tolerance for a given SPO participation level across a range of possible φ values. For each curve, the point at which it drops below 1.0 is the point where uncertainty creeps in as to whether a committee selected with those parameters will tolerate the number of faults indicated on the x-axis.

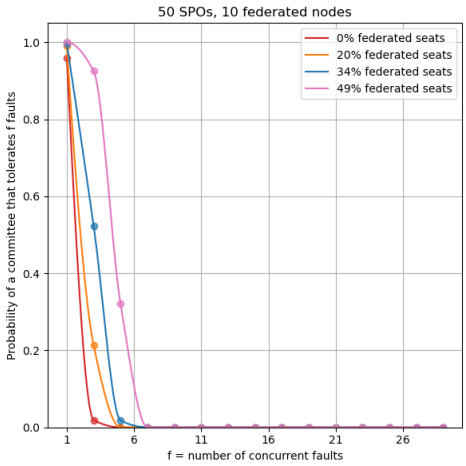

Figure 7: *Fifty SPOs. With such low participation, high federated strength is essential to provide any reasonable degree of fault tolerance. Even with 49% federated, there is a non-zero chance of choosing a commitee with* $f < 4$.

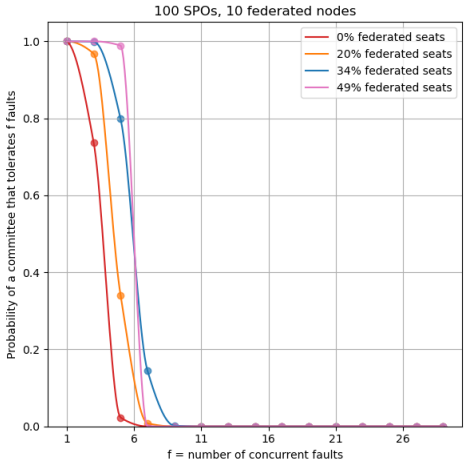

Figure 8: *One hundred SPOs. Even with 100 participants, 49% federated strength still gives the best chance of surviving up to five faults, although 34% has non-zero chance of surviving more than seven faults, where 49% cannot.*

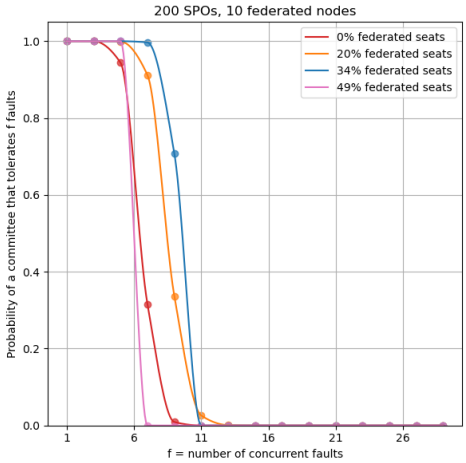

Figure 9: *Two hundred SPOs. Now we see 34% federated strength as the best way to ensure higher fault tolerance. It's conclusively better than 49% at this level of participation.*

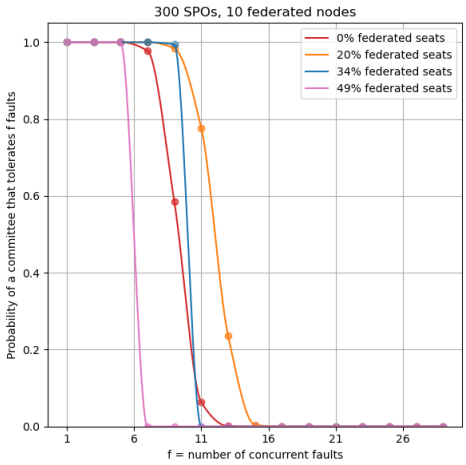

Figure 10: *Three hundred SPOs. Now 20% federated strength is a strong contender. While 34% still wins for maintaining 100% tolerance longer, it plummets to zero chance for fault tolerances where 34% still offers some odds of safety.*

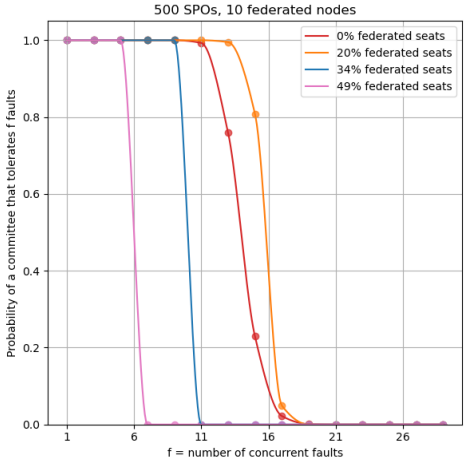

Figure 11: *Five hundred SPOs. Now 20% federated strength is clearly superiod. Higher values of φ have definitely shown their cliff-edge behavior at much lower degrees of fault tolerance.*

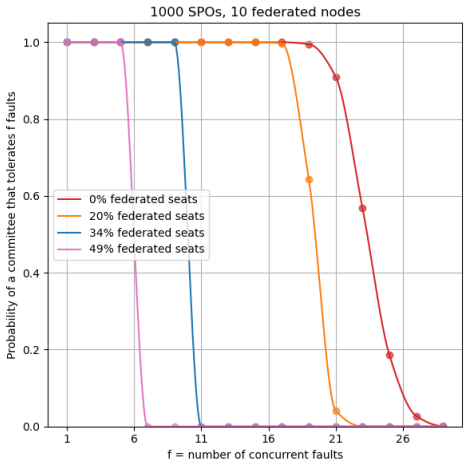

Figure 12: *One thousand SPOs. At this level of participation Ariadne's federated voters do more harm than good. The right strategy is to give them no votes at all.*

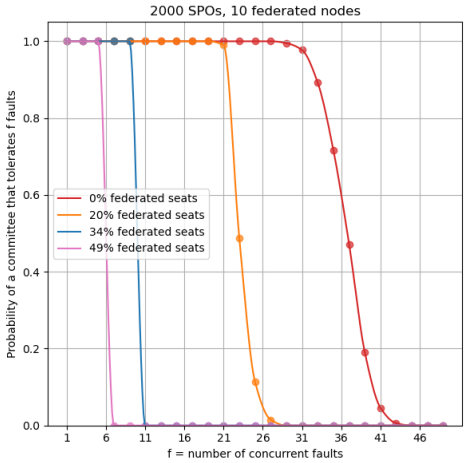

Figure 13: *Two thousand SPOs. (Scale extended up to* $f = 50$ *for this one chart.) In case Midnight achieves this level of participation, the system without federated voters can tolerate up to nearly 30 faults with certainty, with probabilities falling off thereafter. It reaches zero chance of a safe committee when* f *is in the mid 40s.*

6.3 Analysis

These charts tell a consistent story about the benefits of Ariadne. When participation is low, there is benefit in a high degree of federated voting strength. But as the set of SPOs grows, the curves representing higher φ values all exhibit a vertical cliff at some point. As we keep adding SPOs, the curves that employ Ariadne become less and less effective, with the higher φ curves having the worst fault tolerance toward the end.

Figure 14 illustrates why this happens. As the natural pool of SPOs grows, the amount of virtual stake held by the federated committee cannot scale horizontally across more federated nodes, so it effectively scales vertically. In the sorted representation of voting strengths, this shifts the whole block of federated committee members farther and farther to the left. When they become the leading voters, they also become the weakest link in the fault tolerance story.

6.4 Recommendations

These simulations show that Ariadne is not a set-and-forget system. It must be continuously monitored and updated to ensure it is a net benefit to the fault tolerance of the system, or else it can become a significant detriment. Tuning Ariadne's parameters is a governance action, not autonomic. The best way to make the right changes is to track actual participating SPOs and potential non-participating ones, and to run simulations like those used here to ensure the current φ is still advantageous. In the early stages of the network we might have a φ of 34% or even 49%, but we may need to drop to 20% shortly thereafter and we may even go to 0% if we are lucky enough to have thousands of participating SPOs.

7 Toward a Full Model of Finality Risk

A complete model of Midnight's finality risk is outside the scope of this document, but the elements discussed here form one of two pillars for such a model, each of which is largely independent of the other. The rest of this section provides a brief overview of these pillars and how their results can be combined to form a full risk model.

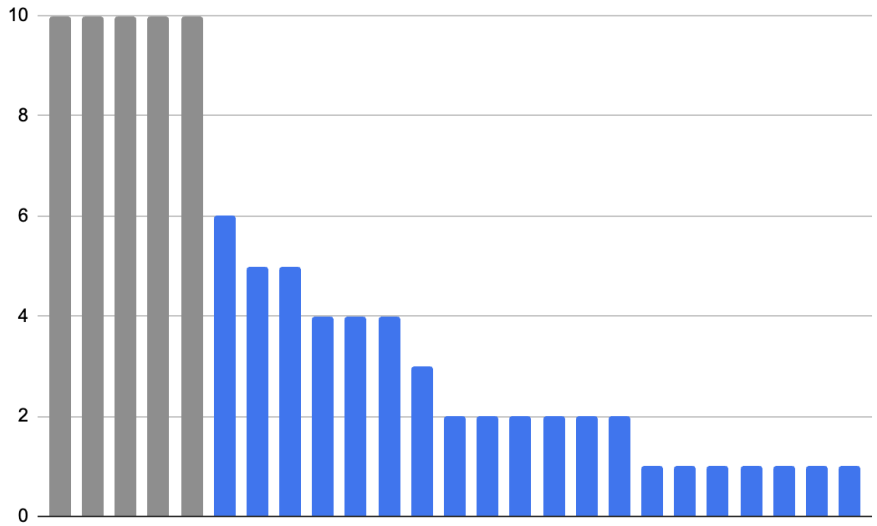

Figure 14: *Rough illustration of the problem of too much federated voting strength when there is high SPO participation. When* φ *is too large and* Γ *is too small, the federated nodes sort to the front of the list and dominate the fault tolerance of the system. Removing them re-normalizes voting strength to the permissionless members, creating higher fault tolerance at higher SPO participation levels.*

7.1 The Risk of a Committee that Cannot Tolerate f Faults

This risk is the focus of the current document. The document provides an approach to analyzing the risk of selecting committees with low fault tolerance, and establishes both the inherent risk and the risk reduction that can be achieved using Ariadne.

7.2 The Risk of f Faults Occurring at Once

This risk is outside the scope of the current document. It's essentially an open-ended problem with a great deal of nuance. We cannot do justice to the topic here, but it might be useful to consider some of the factors to consider in such a model. For most of this discussion we are concerned with improving the *honest uptime* of nodes operated by honest actors; it will be clear when we mean to discuss adversarial actors.

7.2.1 MTBF Abstraction

For a node to become a faulty protocol participant requires either adversarial intent of the failure of some component or interaction among components within the node or the network. Nodes are complex, consisting of many interacting components with their own fault profiles. A deep analysis of all these potential faults can be instructive, but can never be fully complete. Instead, it's common to abstract over root causes by simply measuring the mean time between failure (MTBF) of entire nodes and of their major components. This is much more tractable than trying to analyze the complex interactions of all hardware, software, and network elements.

7.2.2 Detection Time

Byzantine faults are, by nature, hard to detect. It's hard for other nodes to reliably detect a Byzantine fault in another node, but it's also hard for an operator to detect a Byzantine fault in their own node. Since nodes and components suffering Byzantine faults continue to attempt protocol interactions, but do so incorrectly, fault detection requires someone to monitor key performance indicators (KPIs) related to successful protocol interactions. Even when those KPIs indicate an issue, the node operator must determine whether they are at fault or whether they are witnessing faults in other nodes. The longer it takes a node operator to detect their own faults, the

longer they will remain in a faulty state. The average time to detect a fault is called the mean time to detection (MTTD).

7.2.3 Recovery Time

The mean time to recovery (MTTR) is essential in a model meant to predict the risk of concurrent faults. The sum of MTTD and MTTR is the total time the node is not experiencing honest uptime.

If MTBF is much shorter than MTTD+MTTR, it's easy to imagine a system in which all nodes are in recovery at the same time. If MTBF is much greater, it helps reduce the probability of concurrent faults.

7.2.4 DBSync Example

As a Cardano partnerchain, Midnight relies on real-time information from Cardano. Every Midnight node is configured with a DBSync instance that provides a continually updated source of Cardano facts required by the Midnight node for correct operation. If a DB Sync instance fails to keep up with Cardano, missing its real-time targets, then any Midnight nodes that depend on it are faulty. Similarly, if a DB Sync crashes or begins producing incorrect results, its dependent Midnight nodes are faulty. While crash faults of DB Sync seem to be rare, it's not unusual for a DB Sync node to exhibit Byzantine faults such as silently failing its real-time targets.

While it may be possible to replace DBSync with a different technology, current nodes can fail simply because DBSync fails. If the DBSync fault is Byzantine, neither the node software nor the node operator will detect the problem until it manifests as incorrect participation in the Midnight protocol. In the meantime, the node's votes and block-production opportunities are wasted, and preventing the node from validating correct traffic from the rest of the network. Worse, as long as a block producer is running in this failed state, the overall fault tolerance of the network is degraded.

The honest uptime of a node can be improved by a number of approaches, including:

1. Raising DBSync's MTBF (through hardening engineering on that codebase).

2. Reducing the node's MTTD for DBSync.

3. Reducing the MTTR to get the node back into service.
4. Replacing DBSync with a component whose MTBF is higher and/or whose MTTD+MTTR is lower.

One possible mechanism, as an illustration, would be for a set of trusted nodes to operate a small fault-tolerant cluster that periodically votes on the current hash of the important elements of DBSync's state and publishes a signed outcome of their agreed value to Midnight's gossip network. This would allow Midnight nodes to automatically self-check against a known-good fingerprint to greatly reduce MTTD. Such a cluster could also publish full state snapshots of their DBSync instances (using, say `psql_dump`) to provide fast recovery, reducing MTTR.

7.2.5 Abandonment Scenario

Midnight considers all registered SPOs to be active participants in the network. If an SPO registers but later chooses not to participate, they are expected to withdraw their registration. However, if an SPO simply abandons Midnight without de-registering, they will appear faulty whenever selected for a committee. Midnight currently has no mechanism to avoid selecting SPOs who have not participated in block production or voting. As abandoned registrations accumulate, the network's fault tolerance deteriorates.

Addressing this issue is non-trivial, as the protocol must avoid unjustly penalizing SPOs. For instance, if an SPO is temporarily offline due to network issues or a denial-of-service attack, or is experiencing node software failures, should they be suspended? How long must an absence last before suspension is justified?

In the absence of a fully autonomic protocol to manage these edge cases, it may be more practical for Midnight to treat suspension as a governance decision. This would allow the community to define participation standards and adopt rules for suspending SPOs whose behavior does not meet those standards.

7.2.6 Common Cause Analysis

Independent node failures are less of a worry than failures from common causes. If multiple nodes share any common infrastructure or use common operational support, a failure in that infrastructure or support can affect

all those nodes concurrently. In a decentralized system these common-cause faults are extremely hard to predict, because they are the results of individual choices by decentralized actors.

7.2.7 Adversarial Control or Collusion

These are important variants of common-cause faults. In the first case, an adversarial actor corrupts and controls the votes of other participants. In the second case, a number of adversarial actors actively collude to combine their voting power. In either case, such adversarial behaviors fall outside MTBF/MTTR analysis and form a category of their own. The assumption that "*more stake implies greater trust*" must somehow factor into any probability analysis for collusion, but says little about adversarial corruption and control. These are perhaps the hardest risks to model accurately in a decentralized chain.

7.3 Putting it Together

Suppose we can quantify these risks as probabilities, where $\mathrm{P}(\text{not tolerated})_f$ is the probability of selecting a committee that cannot tolerate f concurrent faults, and $\mathrm{P}(\text{occur})_f$ is the probability that f faults might occur simultaneously. If these turn out to be independent probabilities, we can compute the probability of both arising as the product of their probabilities:

$$\mathrm{P}(\text{system fault})_f = \mathrm{P}(\text{not tolerated})_f \times \mathrm{P}(\text{occur})_f$$

There remain some subtleties to explore before we could accept this simple model. Suppose, for example, larger stakepools have lower risk of individual faults. Then the overall model becomes more complicated because the system can generally support more faults among smaller stakepools.

7.4 Telling Risks Apart

It's not always easy to see whether a risk is in one pillar or another, although they are often easily distinguished. Consider a scenario in which the single operator of several large stakepools participates in Midnight for a period of time and then simply stops without de-registering. There is a real risk that the committee-selection protocol will continue to add them to committees despite their lack of activity, since selection does not consider such factors.

The result could be a persistent state in which all their committee members represent perpetual, concurrent faults that the system must carry.

This risk is harder to address than it may seem; it requires a protocol for proposing and challenging claims that a committee member appears inactive, as well as a protocol to allow them to return when they are ready.

We might say that this scenario concerns the risk of choosing a committee that cannot tolerate f faults, since it concerns a blind spot in the committee selection algorithm. But in fact it is primarily about the risk of f concurrent faults because the blind spot has the potential to allow faults to accumulate. The underlying issue is that the candidate has abandoned the protocol, which is a fault.

7.5 Conclusion

This was only a brief overview of the larger risk analysis problem, but it should serve to contextualize the contributions of the current document. Regardless of the specifics of such a combined model, it should be clear that raising the number of faults a committee can tolerate is a significant part of risk management.

A Status & Changes

A.1 Version History

- **1.2.2 (2025-03-24)**:
  - Clarify Byzantine vs. crash faults
  - Explain protocol specification and implementation risks for GRANDPA
  - Explain limitations of flat sampling the SPOs in our simulations
  - Clarify DBSync risks
  - Identify "abandonment" risks
  - Clarify the role of governance in updating consensus parameters
- **1.2.1 (2025-02-19)**: Significant update after early feedback.
  - New title more reflective of the focus of the document within a larger risk model.
  - New section "Toward a Full Model of Finality Risk" that contextualizes this document's contribution within a larger risk analysis.
  - Finished incomplete paragraph concerning participant subsetting.
- **1.01 (2025-02-14)**: Updated page 16 with the correct procedure for choosing a committee in Ariadne.
- **1.0 (2025-02-14)**: Initial version.

A.2 Future Roadmap

- **Revisit Risk Charts**: The charts for inherent and residual risk might be inverted. That is, if we say a chart is about risk, then 100% should be total exposure, while 0% should be no exposure. We should revisit the graphs and wording to be more clear.
- **Incorporate Unused Charts**: Find the right place to include the unused charts in section C.
- **Simulation Details**: Improve description of the residual risk simulation setup.

- **Improved Analysis and Recommendations for Ariadne**: That section is a bit light at the moment.
- **Mitigation**: We have discussed Ariadne as a *preventive* risk, but when something really does go wrong, we need some mitigations and recovery strategies.
- **Health Checks**: Another preventive measure is some basic form of health check we can perform against committee members before adding them to the committee. There are some ideas in the works that need to be added here.
- **Equivocation Penalties**: GRANDPA allows members to detect and respond to equivocation; we should discuss here.
- **Fast Failure Detection**: When well-meaning nodes fail, they may fail in a non-crash manner. We could improve the fault tolerance of the system by reducing the average time required for an operator to learn their node has failed.
- **Fast Failure Recovery**: Similar to fast failure detection, a node that returns more quickly to correct operation is one less failed node for the system to tolerate concurrently.

B Risk Management Framework

This appendix provides a short overview of risk management terminology and procedures.

B.1 Risk Decision-Makers & Appetite

Risk decision-making requires a defined governance structure. Decision-makers establish risk policies, set risk appetite, and enforce risk tolerances. Risk appetite represents the level of risk an organization is willing to accept in pursuit of objectives. Tolerances define acceptable deviations from this threshold.

Risk is not inherently negative. Some risks are necessary to achieve strategic goals. Effective risk management ensures that risks are identified, assessed, and controlled within predefined thresholds.

B.2 Impact vs. Probability (Severity vs. Frequency)

Risk is assessed along two dimensions:

- **Impact (severity):** The consequence of a risk event materializing.
- **Probability (frequency):** The likelihood of the event occurring.

Both dimensions are critical for prioritization. High-impact, low-probability risks may require different strategies than low-impact, high-probability risks. Risk assessments must quantify these attributes where possible to facilitate structured decision-making.

B.3 Risk Event Types & Categories

Risks are classified into structured categories to improve identification, assessment, and response. Classification typically follows a hierarchical structure:

- **High-level categories:** Broad risk domains (e.g., operational, financial, security, compliance).
- **Subcategories:** Specific risk types within each domain (e.g., data breach under security, liquidity risk under financial).

Classification schemes must remain adaptable. New risks may necessitate reclassification, and obsolete categories must be removed to maintain relevance.

B.4 Inherent Risk (Pre-Control Assessment)

Inherent risk is the level of risk in the absence of mitigating controls. It is evaluated along the impact and probability dimensions. Organizations assess inherent risk to:

- Understand baseline exposure.
- Prioritize risk mitigation efforts.
- Establish a benchmark for measuring control effectiveness.

B.5 Risk Controls & Residual Risk

Risk controls are measures that reduce either the impact or probability of a risk event. Controls include technical, procedural, and administrative mechanisms.

Residual risk is the risk that remains after controls are applied. Effective risk management seeks to ensure that residual risk falls within acceptable tolerances.

Control effectiveness is periodically evaluated to verify that controls function as intended. Weak or obsolete controls necessitate adjustment or replacement.

B.6 Monitoring & Adaptive Risk Management

Ongoing monitoring is necessary to maintain an effective risk management framework. This includes:

- **Event tracking:** Risk events are recorded and classified according to the established taxonomy.
- **Adaptive learning:** Risk categories evolve based on observed incidents and emerging threats.
- **Scenario analysis:** Risk exposure is periodically reassessed under varying conditions to identify potential gaps.
- **Control assurance:** Control implementation and effectiveness are routinely evaluated to ensure alignment with risk tolerances.

Monitoring mechanisms ensure that risk management remains an iterative process, continuously adapting to internal and external changes.

C Unused Charts

This section includes some charts and accompanying discussion that are not yet integrated into the main document.

The specific threshold where a given federated voting strength, such as $\varphi = 30\%$, becomes excessive depends on the stake distribution of participating SPOs. It's conceivable, but not in current plans, to automate the selection of a computed φ that brings maximal increase in fault tolerance.

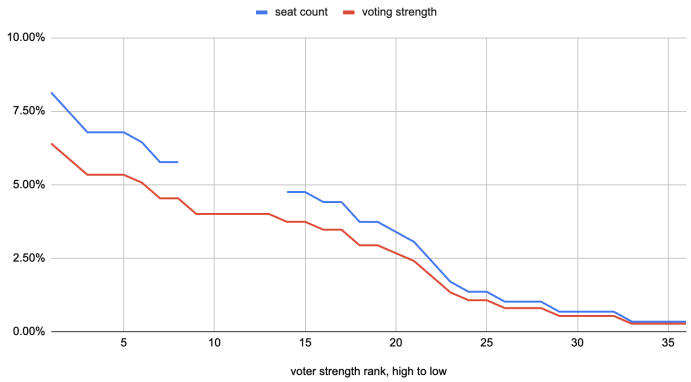

Figure 15: *Scaling effect of federation. This chart shows the scaled voting strength (red) of all committee members after the addition of five federated nodes with 20% of the voting strength. Let* $\mathrm{k} = 300$ *be the committee size. The blue line represents* $\mathit{count}(\mathrm{p})/0.2\mathrm{k}$ *for every unique participant* p *in the permissioness committee, with a gap where the federated nodes will ultimately appear. The red line shows that all nodes are scaled down, but the scaling effect is larger for larger nodes.*

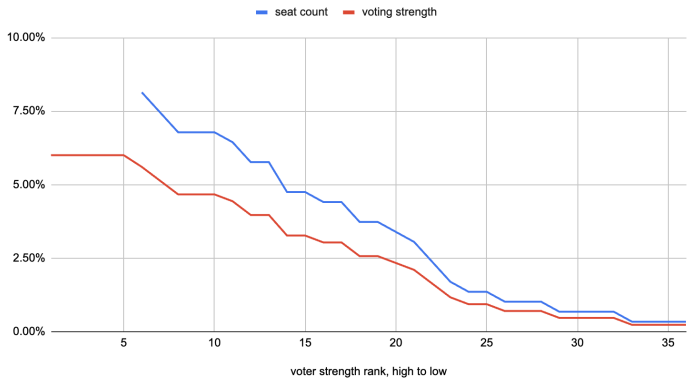

Figure 16: *Similar to Figure 15, but with excessive federated strength of 30%, causing the federated nodes to each have voting strength higher than any SPO. The specific threshold where a given federated voting strength, such as 30%, becomes excessive depends on the stake distribution of participating SPOs.*