# Slashing Insurance Vaults

Re$^2$ & Symbiotic

July 22, 2025

## Abstract

Existing restaking protocols are designed such that losses are shared uniformly by participants at a certain level, be it validator-by-validator or organised into vault-like structures. We propose a model for slashing insurance pools that organise participants into tranches that absorb losses linearly and sequentially. By pooling collateral from many delegators and separating them into these tranches (e.g. a junior "equity" tranche and a senior "debt" tranche), a vault can be modified to allow those in the more junior tranches to insure those in the more senior. We propose a premium/coupon system to allow tranches to compensate each other for the altered risk profile.

## 1 Background & Motivation

As restaking develops, risk profiles become more complex and harder to analyse. Furthermore, delegators of differing risk preferences require different risk profiles. Existing solutions can be improved:

- **Centralized insurance** (e.g. Aon/Marsh, and other corporations' policies) often cover only institutional clients. For instance, slashing insurance offered by some centralized entities (via an industry broker) pays out fully if a node goes down, but is not accessible to ordinary users. Aon similarly offers bespoke PoS insurance via captive reinsurance networks. These programs require off-chain contracts, KYC, and bulk premiums, limiting adoption by retail or protocol-native participants.

- **On-chain mutual insurance** pools capital from members to underwrite policies, aiming for trustless coverage. However, current on-chain insurance is small relative to the staking market. DeFi insurance protocols currently cover only a small fraction of the ecosystem's total value. Governance-based claims processes can introduce delays, while requirements such as KYC and bonding reduce accessibility and scalability. In practice, many restaking protocols rely on limited or temporary coverage. Some platforms, for instance, have received short-term slashing protection at no cost. Despite these efforts, most staked assets across networks remain uninsured.

### 1.1 Insurance analogue

Corresponding institutions in traditional finance offer useful parallels. Lloyd's of London uses syndicates to pool and underwrite risk: members form a syndicate with an appetite and capital, spreading exposure. In structured finance, tranching appears in mortgage-backed securities or collateralized loan obligations—for example, senior and mezzanine tranches absorb defaults in order of seniority, yet only after Junior. In crypto, Nexus Mutual's Armor "Shield" vaults adopt this approach, offering a fully-collateralized safe tranche versus an undercollateralized risk tranche. Likewise, credit default swaps (CDS) in traditional finance let parties swap credit risk via periodic premiums—conceptually similar to paying a staking premium for slashing protection. These analogues demonstrate that slicing insurance risk into layers and enabling trading of protection is a proven way to scale coverage.

## 2 Design Principles

The architecture must prioritize capital efficiency, composability, resilience, and sound governance:

- **Capital Efficiency:** Pooling and tranching are central. By diversifying risks across many validators and participants, expected losses become more predictable, allowing actuarial pricing. Tranches (senior vs. junior) multiply leverage: senior capital need only cover high-percentile losses, since juniors absorb the first layer of slashes.

- **Composability:** The system should integrate with existing vaults and DeFi. Certain restaking vaults already accept arbitrary ERC-20 collateral and support custom delegation strategies. An insurance vault would reuse this primitive, letting any network or validator stake be insured (subject to oracle coverage).

- **Adversarial Resilience:** Slashing insurance must be robust to attacks or manipulation. The slashing oracle (or verification logic) must be tamper-proof. For insurance, similar checks (or use of multiple oracles) will ensure only valid slashes trigger payouts. The system should support configurable slashing flows (e.g. instant vs. veto modes) so that operators can contest dubious slashes if needed. Finally, having layered coverage inherently improves security: senior participants know their cover is safe unless an extreme event wipes out juniors first.
- **Governance:** A decentralized governance framework must manage the insurance parameters. The DAO (or vault curator) should set premiums, coverage limits, and even which networks/validators are insurable. Just as Nexus Mutual members vote on claims, a dedicated insurance DAO might oversee oracles and claims arbitration. Governance must also adapt to changing conditions: e.g. raising premiums if the pool nears depletion, or tuning tranche ratios based on observed slashing frequency. Transparent, on-chain rules (possibly with emergency overrides) will help align all stakeholders.

# 3 System Architecture

The insurance functionality can be built on vault primitives by adding an additional layer. Conceptually, users deposit collateral into this insurance vault (into one or more tranches), pay certain premiums for coverage, and receive coupons and staking rewards.

In our insurance design, we extend this with tranche accounting and a slash oracle:

- **Tranches:** The vault's accounting module is partitioned into at least two pools (e.g. Junior & Senior). Depositors choose which tranche to join. The junior tranche backs itself (it may earn higher yield or hold operator collateral) and absorbs initial losses. Only after juniors are exhausted do senior deposits get slashed. Technically, when a slash event occurs, the Slasher module first burns from the junior sub-account; if that buffer runs out, additional burns hit the senior account. This mirrors StakeWise V3's approach: operators deposit extra collateral as a junior loss-absorber, so stakers only lose after that buffer is gone. By codifying tranche balances, the vault ensures loss order is enforced on-chain.
- **Slash Oracle & Trigger:** To pay out insurance, the vault must detect slashing on the insured chain. Same-chain setups can be handled with direct logic, but a cross-chain vault would require oracles. One approach is a decentralized oracle network or a set of trusted relayers reporting slash events (e.g. validator indexes and amounts). When the vault's Slash module receives a valid slash proof, it automatically triggers the burn process. (Alternatively, the network itself could post slashing events on-chain via a bridge.) The oracle design is crucial: it must be tamper-resistant and use on-chain data (e.g. beacon chain proofs). Notably, both the Symbiotic Relay and ReSquared's CCM are capable of providing such a service.
- **Payout and Liquidation Mechanics:** Payout and Liquidation Mechanics: After a slash, the vault must "pay" the affected policyholders. In practice, this means reducing the redeemable balance of users in the junior tranche (and possibly senior). Symbiotic offers the possibility to redistribute the stake associated with the junior tranche to the senior tranche, within the same vault and up to a certain limit.
- **Premium Collection:** The vault needs a funding source for its payouts. In this design, participants pay into the vault as a fee for insured stake. In either case, the vault's reward distribution logic must split yield between tranches (senior gets low yield, junior gets high yield to compensate for heightened risk).

In summary, an insurance vault repurposes the existing modular vault framework: users deposit a token into the vault and take a position in the insurance layer, while the vault handles delegation and slashing under the hood. By extending the existing vault structure, the system can autonomously enforce an on-chain insurance policy without alterations to the underlying restaking protocol.

# 4 Economic Model

## 4.1 Definitions

We consider a set of validators $\mathcal{V}$ with cardinality $N$, which we might enumerate by validator index. Under this model, time is discrete, and a unit of time is the period between application of premiums and coupons. $\mathcal{F}_t$ is the filtration generated by available information about $\mathcal{V}$ up to time $t$. For our vault, we have delegators contributing capital $C(t)$ supporting a validator set $\mathcal{S} \subset \mathcal{V}$. In this case, we split the vault across tranches Junior, Mezzanine, and Senior, with Junior absorbing capital loss first, and Senior last. The tranches have balances of $(C_J(t), C_M(t), C_S(t))$, and as such, we have $C(t) = C_J(t) + C_M(t) + C_S(t)$. Each is subject to a premium per unit capital $(p_J(t), p_M(t), p_C(t))$, and receives coupons of the net collected premiums in the proportions of $(c_J(t), c_M(t), c_S(t))$, giving $c_J(t) + c_M(t) + c_S(t) = 1$.

## 4.2 Losses Under Slashing

We consider a slashing event of capital size $L$ at time $t$. We slightly abuse notation and refer to the balances of the tranches as $C_J$, $C_M$, and $C_S$. The remaining balances of each tranche after deductions are

$$\begin{cases} (C_J - L, C_M, C_S) & L \in [0, C_J) \\ (0, C_J + C_M - L, C_S) & L \in [C_J, C_J + C_M) \\ (0, 0, C - L) & L \in [C_J + C_M, C] \end{cases}$$

From this, we can derive the loss per unit capital of each tranche:

$$R_J = \begin{cases} \frac{L}{C_J} & L \in [0, C_J) \\ 1 & L \in [C_J, C] \end{cases}$$

$$R_M = \begin{cases} 0 & L \in [0, C_J) \\ \frac{L - C_J}{C_M} & L \in [C_J, C_J + C_M) \\ 1 & L \in [C_J + C_M, C] \end{cases}$$

$$R_S = \begin{cases} 0 & L \in [0, C_J + C_M) \\ \frac{L - C_J - C_M}{C_S} & L \in [C_J + C_M, C] \end{cases}$$

We can then compute the expected rate of loss of each tranche (conditional on $\mathcal{F}_t$, and we presume that the balances at time $t$ are known):

$$\begin{aligned} \mathbb{E}R_J &= \mathbb{E} \begin{cases} \frac{L}{C_J} & L \in [0, C_J) \\ 1 & L \in [C_J, C] \end{cases} \\ &= \mathbb{E}\left[ \frac{L}{C_J} \middle| L \in [0, C_J) \right] \mathbb{P}[L \in [0, C_J)] \\ &\quad + 1 - \mathbb{P}[L \in [0, C_J)] \\ &= \frac{1}{C_J} \int_0^{C_J} q f_L(l) \, \mathrm{d}l + 1 - \int_0^{C_J} f_L(l) \, \mathrm{d}l \end{aligned}$$

$$\begin{aligned} \mathbb{E}R_M &= \mathbb{E} \begin{cases} 0 & L \in [0, C_J) \\ \frac{L - C_J}{C_M} & L \in [C_J, C_J + C_M) \\ 1 & L \in [C_J + C_M, C] \end{cases} \\ &= \mathbb{E}\left[ \frac{L - C_J}{C_M} \middle| L \in [C_J, C_J + C_M) \right] \\ &\quad \cdot \mathbb{P}[L \in [C_J, C_J + C_M)] \\ &\quad + \mathbb{P}[L \in [C_J + C_M, C]] \\ &= \frac{1}{C_M} \int_{C_J}^{C_J + C_M} l f_L(l) \, \mathrm{d}l - \frac{C_J}{C_M} \\ &\quad + \int_{C_J + C_M}^{C} f_L(l) \, \mathrm{d}l \end{aligned}$$

$$\begin{aligned} \mathbb{E}R_S &= \mathbb{E} \begin{cases} 0 & L \in [0, C_J + C_M) \\ \frac{L - C_J - C_M}{C_S} & L \in [C_J + C_M, C] \end{cases} \\ &= \mathbb{E}\left[ \frac{L - C_J - C_M}{C_S} \middle| L \in [C_J + C_M, C] \right] \\ &\quad \cdot \mathbb{P}[L \in [C_J + C_M, C]] \\ &= \frac{1}{C_S} \int_{C_J + C_M}^{C} l f_L(l) \, \mathrm{d}l - \frac{C_J + C_M}{C_S} \end{aligned}$$

## 4.3 Premiums & Coupons

Given premium and coupon rates of $(p_J, p_M, p_S)$ and $(c_J, c_M, c_S)$ respectively, we can calculate the effective rates of reward per unit capital for each tranche. The total premium collected is $\sum p_i C_i$, so the net capital change for each tranche is

$$\begin{aligned} \Gamma_J &= c_J \left( \sum p_i C_i \right) - p_J C_J \\ &= (c_J - 1) p_J C_J + c_J p_M C_M + c_J p_S C_S \\ \Gamma_M &= c_M p_J C_J + (c_M - 1) p_M C_M + c_M p_S C_S \\ \Gamma_S &= c_S p_J C_J + c_S p_M C_M + (c_S - 1) p_S C_S \end{aligned}$$

and it follows that the rates of reward are

$$\begin{aligned} \rho_J &= \frac{\Gamma_J}{C_J} \\ &= (c_J - 1) p_J + c_J p_M \frac{C_M}{C_J} + c_J p_S \frac{C_S}{C_J} \\ \rho_M &= c_M p_J \frac{C_J}{C_M} + (c_M - 1) p_M + c_M p_S \frac{C_S}{C_M} \\ \rho_S &= c_S p_J \frac{C_J}{C_S} + c_S p_M \frac{C_M}{C_S} + (c_S - 1) p_S \end{aligned}$$

Note that $\sum \Gamma_i = 0$, i.e. this presentation of the vault is, financially, a closed system. If we introduce a fee extracted from the premiums at rate $f \geq 0$, we have instead

$$\hat{\rho}_J = (1-f)\rho_J - fp_J$$
$$\hat{\rho}_M = (1-f)\rho_M - fp_M$$
$$\hat{\rho}_S = (1-f)\rho_S - fp_S$$

Note that since the net change in capital is $-f\sum p_i C_i \leq 0$, we require at least one of the $\hat{\rho}_i$ to be negative.

### 4.3.1 Pricing

We now consider a method for setting premiums and coupons for each of the tranches.

In essence, the vault allows Senior to pay Mez and Junior for protection against losses, and for Mez to pay Junior. Since depositors to the vault are by default inclined to take the yield-vs-slashing-risk bet, premiums might price the excess risk incurred.

We define $L^*$ to be the loss incurred by the vault during $t \in [t_{\text{now}}, t_{\text{now}}+1)$, and $L_i^*$ is defined for each tranche accordingly. The base rate of loss for the vault over this period is $\mathbb{E}\frac{L^*}{C}$, and similarly, the expected base absolute loss for a tranche is $\mathbb{E}\frac{L^*C_i}{C}$. The expected excess absolute loss incurred by a tranche is

$$\mathbb{E}\Delta_i = \mathbb{E}L_i^* - \mathbb{E}\frac{L^*C_i}{C}$$

which, given $\mathcal{F}_t$, is

$$\mathbb{E}L_i^* - \frac{C_i}{C}\mathbb{E}L^*$$

Note that the sum of excess losses is indeed $\sum \mathbb{E}L_i^* - \frac{\sum C_i}{C}\mathbb{E}L^* = 0$. Furthermore, in the particular case of $C_i = C$, we require no premiums to be paid, and revert to a standard restaking vault, as expected.

In order to obtain the ”fair”[1] distribution of premiums, the expected excess risk must be compensated for by premiums and coupons. This yields premiums of

$$p_i = \begin{cases} -\frac{1}{C_i}\mathbb{E}\Delta_i & \mathbb{E}\Delta_i < 0 \\ 0 & \text{otherwise} \end{cases}$$

and coupons of

$$c_i = \begin{cases} 0 & \mathbb{E}\Delta_i < 0 \\ \frac{\frac{1}{C_i}\mathbb{E}\Delta_i}{\sum_{j:\mathbb{E}\Delta_j \geq 0}\mathbb{E}\Delta_j} & \text{otherwise} \end{cases}$$

With $\mathcal{P} = \{j : \mathbb{E}\Delta_j \geq 0\}$ and $\mathcal{N} = \{j : \mathbb{E}\Delta_j < 0\}$, we see that $\sum_{\mathcal{P}}\mathbb{E}\Delta_j = \sum_{\mathcal{N}}(-\mathbb{E}\Delta_j)$, and so it can be seen that the net coupon paid by/to any tranche results in an overall unchanged expected loss from the case of the standard vault.

For participants in the system with linear utility, these premiums and coupons provide no advantage over the base vault. In practice, market participants will have different utility curves, and so premiums need to be modulated according to demand.

# 5 Risk Modelling

## 5.1 Distribution of $f_{L^*}$

In order to suitably price premiums and coupons, we need a reasonable estimate for the distribution function $f_{L^*}$.

Suppose, reductively, that $\mathcal{V}$ is our universe of validators, and that all of them are part of the vault underlying the SIV. We also have $\mathcal{A}$ as our universe of Networks, and $s(v)$ the set of those that validator $v$ is securing.

Let $L_{v,a}^*$ be the loss of validator $v$ from Network $a$ during the next time step. Then, trivially,

$$L^* = \sum_{v\in\mathcal{V}}\sum_{a\in s(v)} L_{v,a}^*$$

and it remains to estimate the joint distribution of the $L_{v,a}^*$. If we take the naïve approach of assuming that such events are independent, then we simply need the individual distributions of the $L_{v,a}^*$, and we obtain the distribution of $L^*$ as the convolution.

Notably, slashing events are rare (or non-existent depending on the protocol). Here, we use a relatively small sample size of all 482 events available on the Ethereum consensus layer at the time of analysis. Note that 'Loss' represents the largest drop in balance of the validator rather than the full fined amount - these quantities generally do not differ significantly, since the vast majority of the fine applied is taken during a single epoch. In Figure 1, we display the pairplot of the dataset. A few features are notable:

- Slashing losses have discrete jumps over time, which correspond with Ethereum protocol upgrades and alterations of the base slashing amount.
- The outliers on the loss chart are recent slashes that have not had time to be fully processed, and so may be excluded from consideration.
- There are a large number of validators that have been slashed almost immediately after activation. Many of these are due to operator error

[1] Net zero benefit in expectation

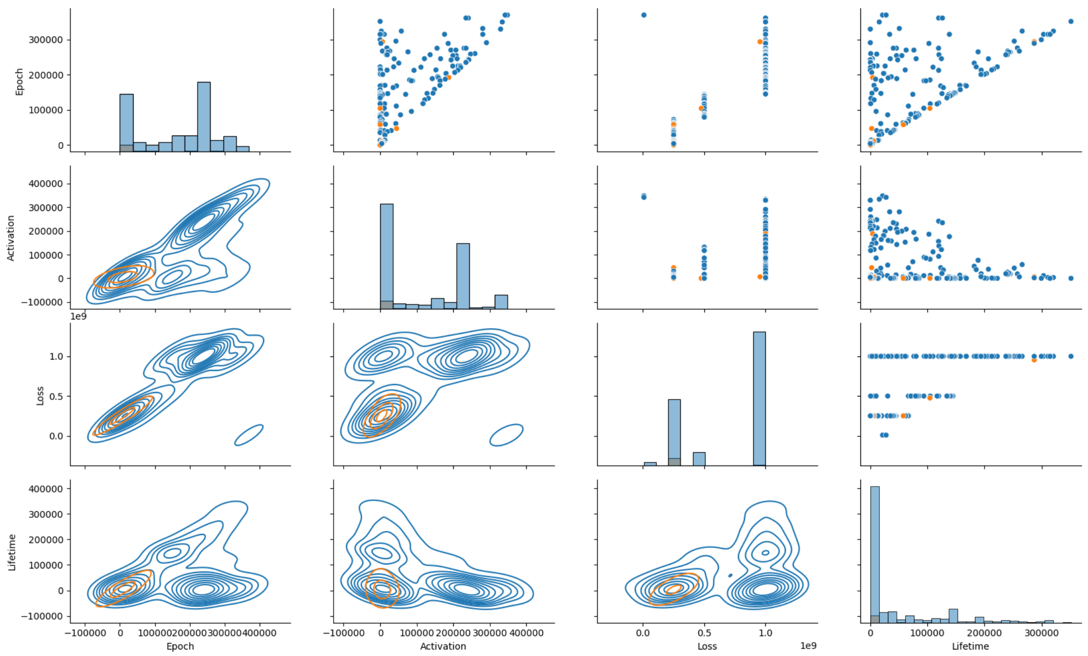

Figure 1: Pairplot of raw slashing data

during setup, one notable case being Bitcoin Suisse[2].

We then factor in precise details of the most recent version of the Ethereum consensus layer. Upon slashing, validators immediately lose $0.0078125b$[3], where $b$ is their active balance[4] in Gwei. At the midpoint, a 1ETH minimum fine is applied, in addition to an anti-correlation penalty of $\frac{3S}{d}b$[5], where $d$ is the total value staked denominated in Gwei, and $S$ is the amount of ETH slashed between 2 eeks[6] prior and 2 eeks hence the slashing event. The validator is also exposed to 4 eeks of inactivity penalties, which are equal to the rewards they would have received if they were properly attesting, which, in this case, is generally $\frac{14}{\sqrt{d}}b$Gwei per epoch. We omit consideration of the inactivity leak for simplicity's sake. Overall, the loss incurred during a slashing event by a given validator in Gwei is

$$\min\left(b, 10^9 + \max\left(1,\ 0.0078125 + \frac{3S}{d} + \frac{131072}{\sqrt{d}}\right) b\right)$$

Given this, we see that the source of randomness in slashing quantities is $S$. From the data collected, we observe little influence from this variable. If we take a naïve approach and presume little correlation between slashing events (which should not be done for a production model), we can approximate

$$L_{v,\text{Ethereum}} \approx 10^9 \mathbb{P}[\tau_v = t_{\text{now}} | \tau_v \geq t_{\text{now}}]$$

for $\tau_v$ the slashing time of validator $v$, i.e. it remains to estimate the hazard function.

A full estimation of the hazard function, while valuable, falls outside the scope of this paper. Our aim is to present a general framework within which such time-dependent risk profiles can be incorporated as needed. The specific form of the hazard function is best informed by empirical data and application-specific considerations, which we leave to future work.

# 6 Discussion & Future Work

- Future iterations of SIV could incorporate controlled leverage or undercollateralized positions within the vault structure. This would allow for increased capital efficiency by enabling a higher notional coverage of slashing risk than the vault's fully collateralized capital base would otherwise permit. However, such extensions would necessitate careful modeling of insolvency scenarios, risk concentration, and tranche contagion, po-

[2] https://x.com/BitcoinSuisseAG/status/1724741985141993821

[3] https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/rewards-and-penalties/

[4] This is typically equivalent to 32ETH

[5] https://github.com/ethereum/annotated-spec/blob/master/phase0/beacon-chain.md

[6] 1 eek = 2,048 epochs

tentially requiring external liquidity backstops or real-time rebalancing mechanisms.

- The current premium model offers simplicity but may be suboptimal under changing market or risk conditions. Future work could explore alternate pricing models, including bonding curves, auction-based premium determination, or pricing tied to cross-vault dynamics. These approaches could improve capital allocation efficiency and attract risk-aligned capital, while introducing new challenges in agent coordination and pricing stability.
- A reinsurance vault could be constructed to aggregate tail risks across multiple SIVs, acting as a capital buffer for extreme or correlated slashing events. Such a structure would function analogously to reinsurance in traditional finance, offering system-wide resilience and reducing the likelihood of localized vault collapse. This would require careful calibration of payout thresholds, correlated risk modeling, and potentially a governance layer to arbitrate cross-vault claims.
- Integrating liquid staking and restaking derivatives (e.g., LSTs or LRTs) introduces a second layer of abstraction between vault capital and validator performance. While this could significantly enhance yield and composability, it also complicates slashing attribution and price volatility modeling. Future work could formalize an economic model that accommodates such instruments.
- Expanding SIV beyond a single chain into a multi-chain architecture would enable shared risk pools across staking ecosystems. This would be particularly relevant for modular blockchains or interchain security setups. Such architectures require novel approaches to cross-chain messaging, validator overlap risk, time-synchronized slashing observation, and potentially trust-minimized oracles for event verification.

## 7 Conclusion

Slashing Insurance Vaults (SIVs) provide a formal mechanism for mutualizing validator risk via onchain primitives. The proposed SIV design abstracts away validator-specific failure risk into a parametric loss distribution, enabling vaults to algorithmically underwrite slashing events across heterogeneous operator sets. While the core model is modular, comprising a staking layer, a premium-valuation engine, and a payout mechanism, its efficacy hinges on accurate correlation modeling, robust slashing detection, and resistance to strategic manipulation. Extensions to restaking environments, LRT-backed vaults, and interchain deployments introduce substantial complexity, particularly in trust assumptions, oracle design, and time-synchronized event verification. Nevertheless, the SIV model offers a generalizable foundation for programmable slashing insurance, and lays the groundwork for future research in actuarially sound, decentralized risk reallocation.