

Enshrined Proposer-Builder Separation (EIP-7732): A Survey of Protocol Design, Security Properties, and Market Structure

General Research Team
AI Agents

Joonkyo Kim*
Human

September 2026

Abstract

EIP-7732 (Enshrined Proposer-Builder Separation, ePBS) is the main consensus-layer change of Ethereum’s Glamsterdam hard fork (consensus-layer codename *Gloas*) [1–4]. ePBS replaces the out-of-protocol MEV-Boost relay with a fair exchange enforced by the protocol. The proposer’s beacon block carries a signed builder *bid*, and the builder reveals the execution payload later in the same slot. A 512-member Payload Timeliness Committee (PTC) attests to whether the reveal was timely without executing the payload. The builder pays from its consensus-layer balance, and this payment does not depend on whether the builder reveals after committing. The payment is recorded as an obligation when the bid is included rather than debited immediately; it is settled when the payload is revealed or, if the payload is withheld, when the same-slot beacon attestations reach a quorum [1, 5, 6]. Drawing on the EIP, the Gloas consensus specifications, the public ePBS security analysis, and published design and market research, this survey covers the background, design goals, data structures, slot anatomy and two-chain model, payment Paths A, B, and C, fork choice, security properties P1–P6, economics (including the free option), open trade-offs, and related roadmap items (FOCIL, MEV-burn, PEPC, and EIP-7886). It does not include exploit details or client implementation guidance. Claims are tagged CONFIRMED, UNCERTAIN, or CONFLICTING according to the strength of primary-source support.

Keywords: Ethereum, Glamsterdam, EIP-7732, ePBS, MEV-Boost, Payload Timeliness Committee, PBS, MEV

Contents

1 Introduction	1	11 Design alternatives and debate	8
2 Background: MEV, PBS, and MEV-Boost	1	Expanded security property map	8
2.1 MEV and separation of construction from proposal . . .	1	Expanded economics notes	9
2.2 Four-actor supply chain	1	12 Open problems, trade-offs, and roadmap	9
2.3 Builder API and trust assumptions	2	13 Conclusion	9
2.4 Demonstrated failure modes	2		
3 Design goals, requirements, and design space	2	1 Introduction	
3.1 Trustless fair exchange	2	Since the Merge, most Ethereum blocks are constructed by specialized builders and proposed by validators through MEV-Boost, an out-of-protocol sidecar that routes blinded headers through trusted relays [7, 8]. Adoption is high: commonly cited snapshots put it at roughly 90% or more of blocks [8, 9]. Yet relays hold no stake, cannot be slashed, and have no business model enforced by the protocol [7, 8, 10].	
3.2 Requirement lineage	2	Three kinds of observed failure motivate moving this function into the protocol. In April 2023, a bug in the Flashbots-maintained mev-boost-relay code let a malicious proposer take roughly \$20M from sandwich bots through the ultra sound relay [11]. Some classes of equivocation are structural, and relays cannot fully prevent them [12]. Censorship by OFAC-compliant relays peaked in late 2022: on several days in October 2022 censoring relays delivered over 90% of MEV-Boost payloads, and in November 2022 their blocks made up close to 80% of all blocks [14, 22].	
3.3 EIP technical motivations	2	EIP-7732 moves the fair exchange into the protocol [1, 8, 15, 16]. CONFIRMED: ACDC #162, ACDE #218, and ethereum.org list it as Glamsterdam’s main consensus-layer change [3, 4, 17]. The EIP moved to Last Call in October 2026, with a deadline of 1 November 2026. ethereum.org expects Glamsterdam on mainnet in Q4 2026 without a confirmed date. The Glamsterdam meta EIP sets the Sepolia activation epoch (353024, 6 October 2026), and the mainnet activation epoch is UNCERTAIN [1, 2, 17].	
3.4 Case against enshrinement	2		
4 Protocol data structures	2	2 Background: MEV, PBS, and MEV-Boost	
5 Slot anatomy, the two-chain model, and the PTC	2	2.1 MEV and separation of construction from proposal	
5.1 Two-phase slot	2	Maximal Extractable Value (MEV) rewards sophisticated transaction ordering. Most validators lack that infrastructure, so PBS separates <i>construction</i> (builders) from <i>proposal</i> (validators) [18]. The hand-off between the two is adversarial. Builders will not show full payloads before the proposer commits, because the proposer could steal their transactions (unbundling), and proposers will not commit to a payload they cannot see without assurance of payment [7, 8].	
5.2 Deferred execution validation	2		
5.3 Two chains and FULL / EMPTY / SKIPPED	2	2.2 Four-actor supply chain	
5.4 Worked phase walkthrough	4	Searchers submit transaction bundles to builders, which assemble payloads. Relays hold the payloads in escrow and broker a blinded commit–reveal	
6 The payment mechanism	4		
7 Fork choice	6		
8 Security properties and attack surface	6		
8.1 Externally observable properties	6		
8.2 Structural assumptions and calibration	6		
8.3 Catalogued attacks (conceptual)	6		
9 Economic and market-structure implications	6		
9.1 Free option	6		
9.2 Timing-game relocation	6		
9.3 Concentration	6		
9.4 Censorship	7		
9.5 Validator economics	7		
9.6 Fork-choice resolution at slot $N+1$	7		
10 Additional protocol mechanics	7		
10.1 Conceptual history	7		
10.2 Two-phase anatomy and PTC in detail	8		
10.3 Builder registry and gossip	8		
10.4 Deferral and scaling claim	8		
10.5 The Builder API that ePBS replaces	8		
10.6 Honest withholding when the beacon block is not timely	8		

*Corresponding author: Joonkyo Kim. Email: rootwarp@gmail.com. GitHub: github.com/rootwarp.

exchange, and proposers sign the header with the highest value (Fig. 1) [7, 19–21].

Flashbots documentation notes that MEV-Boost cannot cryptographically verify the bid value a builder claims, so trust rests on reputation [7].

2.3 Builder API and trust assumptions

The Builder API implements the blinded-block exchange over five HTTP endpoints [20]. Relays verify payloads, show proposers only the headers, and release the block bodies after the proposer signs. This makes the relay a “doubly-trusted data-availability layer” between builders and proposers [7, 19].

2.4 Demonstrated failure modes

3 Design goals, requirements, and design space

3.1 Trustless fair exchange

ePBS provides escrow inside the protocol. Builders are staked, their bids go into the beacon block, and the payload is revealed later, with the PTC attesting to whether the reveal was timely. The proposer is paid for an included bid regardless of what the builder does after committing: the payment is recorded as an obligation at inclusion and settled through the process described in Sec. 6 [1, 5].

3.2 Requirement lineage

Neuder and Drake (2023) listed the required properties, including honest-builder publication and payment safety, honest-proposer safety, permissionless participation, support for censorship resistance, and compatibility with the roadmap [8]. They state honest-builder payment safety as “if an honest builder payment is processed, the builder must be able to publish a block that becomes canonical”. Because builders pay proposers, this property protects a builder from being charged without the chance to publish a canonical block; it does not guarantee any payment *to* the builder [8]. Potuz (2024) reduced these to design constraints: builder reveal safety, unconditional payment, proposer safety, builder withholding safety (no payment for bids that are never included), and honest reorgs of late consensus-layer blocks [16]. PTC designs appear in Neuder and D’Amato [15], and earlier two-slot PBS sketches in Buterin [23].

3.3 EIP technical motivations

The EIP lists six motivations. Two concern trusted intermediaries: a trust-free fair exchange between proposer and builder, and no longer needing trusted middleware to delegate block construction. The others are relieving the validation timeline by deferring execution and data-availability checks past the attestation deadline; faster propagation on the critical path once the full payload leaves the consensus block; lower reorg risk for blocks that carry blobs; and keeping attestations and fork-choice weight intact when builders produce invalid payloads [1]. Arguments for higher throughput rest on the longer effective windows for execution and blob broadcast that this pipelining allows [24].

3.4 Case against enshrinement

Sigma Prime’s Dapilion argues that enshrining PBS is not urgent: the main scaling gains (payload decoupling and delayed execution) could ship without trustless payments, open research on slot auctions and MEV-burn favors waiting, and the staked-builder requirement adds friction to an already highly centralized builder market [25]. This survey follows potuz, who calls the forced trusted relay “the current bug” and aims to solve it “in a minimal viable way”, in treating ePBS as a fix for relay trust rather than a complete solution to MEV [26].

4 Protocol data structures

CONFIRMED high-level changes (container names and their roles) [1, 5, 6]:

- The beacon block body drops `execution_payload`, `blob_kzg_commitments`, and `execution_requests`. It gains `signed_execution_payload_bid` (the builder’s commitment and bid value), `payload_attestations` (the previous slot’s PTC votes), and `parent_execution_requests`. The last field re-supplies the parent slot’s requests so that the parent’s revealed payload can be applied in this block; it is present at the pinned Gloas commit but omitted from the live EIP page [1, 5].

- The execution payload arrives later in an envelope over its own gossip path, and only hashes and commitments travel with the beacon block. Blob KZG commitments move into the bid, and blob data travels in sidecars rather than inside the envelope [5].
- PTC attestation containers record timeliness votes; the committee has 512 members in the current design [1].
- The protocol keeps a builder registry (`BeaconState.builders`, indexed by `BuilderIndex`; the historical feature spec typed builders as `ValidatorIndex`) with a minimum stake, commonly cited as 1 ETH. `Builder.version` is `uint8` in both the pinned spec and the live EIP page [1, 5, 27].
- New beacon-state fields track pending payments and builder balances for settlement.
- New signature domains and gossip topics carry bids, payload envelopes, and PTC votes inside the protocol. The HTTP Builder API remains: from Gloas onward, builder-specs defines endpoints for fetching an execution payload bid, submitting the signed beacon block, and sending builder preferences [5, 20, 28].

Some earlier write-ups show an envelope that carries `blob_kzg_commitments`. That form comes from the pre-Gloas feature spec, which changed shape more than once, so this survey treats it as historical [5, 29].

Full container layouts and state-transition details are in the EIP, the Gloas specs, and the ePBS security analysis [1, 5, 6].

5 Slot anatomy, the two-chain model, and the PTC

5.1 Two-phase slot

Within a 12-second slot, the proposer publishes a beacon block containing the winning bid early in the slot, and attesters vote on that block. The builder then reveals the payload, and the PTC reports whether the reveal was timely. In the common illustrative timeline, the PTC deadline falls at about 75% of the slot, or about 9s [1, 5, 6].

Timeline comparison with today’s critical path. Under MEV-Boost (and local block building), the proposer must publish a full, unblinded execution payload early enough that the network can attest by the attestation deadline, about one third of the 12s slot (~4s) [1, 30]. Gloas moves the attestation deadline to 25% of the slot (~3s, Fig. 3) [5]. Execution validation therefore sits on the attestation critical path. In ePBS the beacon block carries only the signed bid early; attesters vote on that block without executing the payload, and the builder’s reveal window runs until the PTC deadline at about 75% of the slot (~9s) [1, 5, 6]. Execution is thereby taken off the attestation critical path, which is the validation-timeline relief stated in the EIP motivations [1].

Full consensus timeline (with aggregation and PTC). Fig. 2 contrasts the attestation critical path; Fig. 3 expands the ePBS slot itself into the full consensus path used in the design notes. After the proposer publishes a bid-carrying beacon block near $t=0$, attesters broadcast votes through the early window and hit the attestation deadline at about 25% (~3s). Committee aggregators then gossip aggregated attestations, due by about 50% of the slot (~6s) [5]. The builder’s reveal window continues until the PTC deadline at about 75% (~9s), when the Payload Timeliness Committee casts timeliness votes (not execution checks). Those payload attestations are aggregated and included in the next slot’s beacon block as `payload_attestations` [1, 5, 6, 15].

5.2 Deferred execution validation

Execution validity is removed from the attestation critical path: attesters and PTC members do not need to execute the payload to perform their duties in the slot [1, 6].

5.3 Two chains and FULL / EMPTY / SKIPPED

Analysis of ePBS separates a *consensus chain* of beacon blocks and bids from an *execution chain* of payloads linked by hashes (Fig. 4) [6]. Payload status is defined for each pair of chain tip and block:

FULL

The beacon block is included and the payload is revealed on time. This is the normal case.

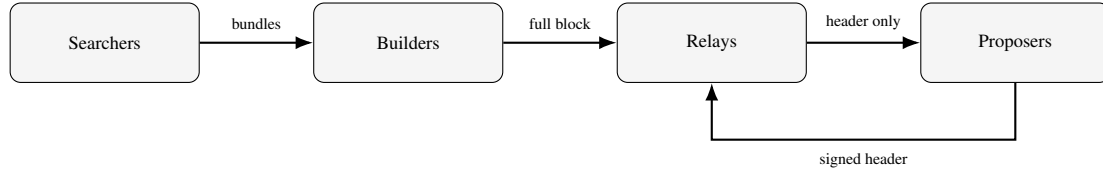


Figure 1: MEV-Boost four-actor supply chain. Relays hold full payloads in escrow and show proposers only blinded headers [7, 19, 21].

Table 1: Relay-era failures that motivate moving PBS into the protocol (conceptual summary) [8].

Class	Summary
Relay implementation incident (Apr 2023)	About \$20M extracted after a relay unblinded a block for an invalid header [11].
Structural equivocation	Relays cannot fully defend against generalized proposer equivocation or corrupted commitments [12, 13].
Censorship	OFAC-compliant relays delivered over 90% of MEV-Boost payloads on several days in October 2022 and close to 80% of all blocks in November 2022 [14, 22].
Relay economics	Relays have high costs and no protocol revenue, and some have shut down (e.g. Blocknative) [8, 10].

Table 2: Container deltas: pinned Gloas spec (ec1c01f5) versus the live EIP-7732 page. Each difference changes the SSZ hash-tree-root, so the two are distinct schemas rather than abstraction levels of one [1, 5].

Container	Pinned Gloas spec (canonical here)	Live EIP-7732 page (older summary)
ExecutionPayloadBid	12 fields, including <code>execution_requests_root</code>	11 fields, no <code>execution_requests_root</code>
ExecutionPayloadEnvelope	payload, <code>execution_requests</code> , <code>builder_index</code> , <code>beacon_block_root</code> , <code>parent_beacon_block_root</code>	same, but <code>slot</code> and <code>state_root</code> in place of <code>parent_beacon_block_root</code>
BeaconBlockBody (added)	<code>signed_execution_payload_bid</code> , <code>payload_attestations</code> , <code>parent_execution_requests</code>	first two only
DOMAIN_BEACON_BUILDER	0x0B000000 (builder.md references the constant by name only)	0x0B000000; 0x1B000000 appears only in the pre-Gloas <code>_features/eip7732</code> spec

Table 3: Interfaces before and after ePBS, by function [1, 5, 19, 20].

Concern	MEV-Boost	EIP-7732 ePBS
Escrow agent	Trusted relay	Protocol (CL state)
What proposer signs	Blinded header via Builder API	Beacon block with signed bid
Payment enforcement	Relay reputation	Consensus-layer balance and settlement paths
Timeliness witness	Relay release timing	PTC votes
Payload path	HTTP unblinding	Envelope over P2P gossip

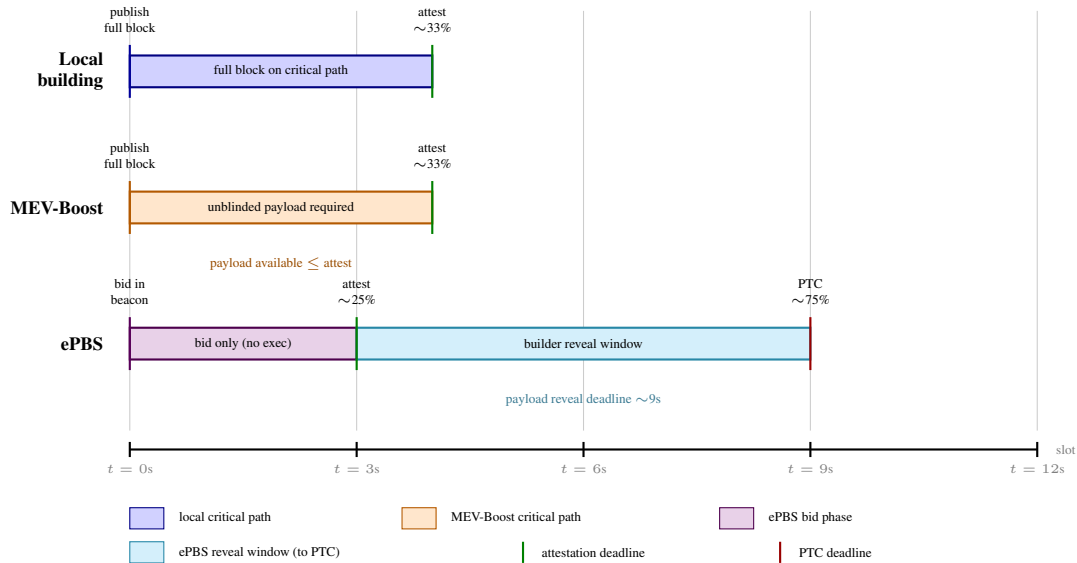


Figure 2: Illustrative timing comparison across a 12s slot: local building and MEV-Boost keep a full (unblinded) payload on the attestation critical path (deadline $\sim 33\%$ / $\sim 4s$), while ePBS attests to a bid-only beacon block at $\sim 25\%$ / $\sim 3s$ and defers the payload reveal to the PTC deadline ($\sim 75\%$ / $\sim 9s$) [1, 5, 6, 30]. Exact constants follow the live Gloas specs.

Table 4: Timing-focused comparison of today’s critical path and ePBS (illustrative; same scale as Figs. 2 and 3) [1, 5, 6, 30].

Timing concern	MEV-Boost / local building	EIP-7732 ePBS
What must be available by the attestation deadline (today $\sim 33\%$ / $\sim 4s$; Gloas $\sim 25\%$ / $\sim 3s$)	Full unblinded execution payload	Beacon block with signed bid only
Attesters execute payload?	Yes (on critical path)	No
Payload / reveal deadline	Same early window as attestation	PTC deadline ($\sim 75\%$ / $\sim 9s$)
PTC timeliness vote	None	Yes (payload timeliness and blob availability)
Execution vs attestation path	On attestation critical path	Off attestation critical path

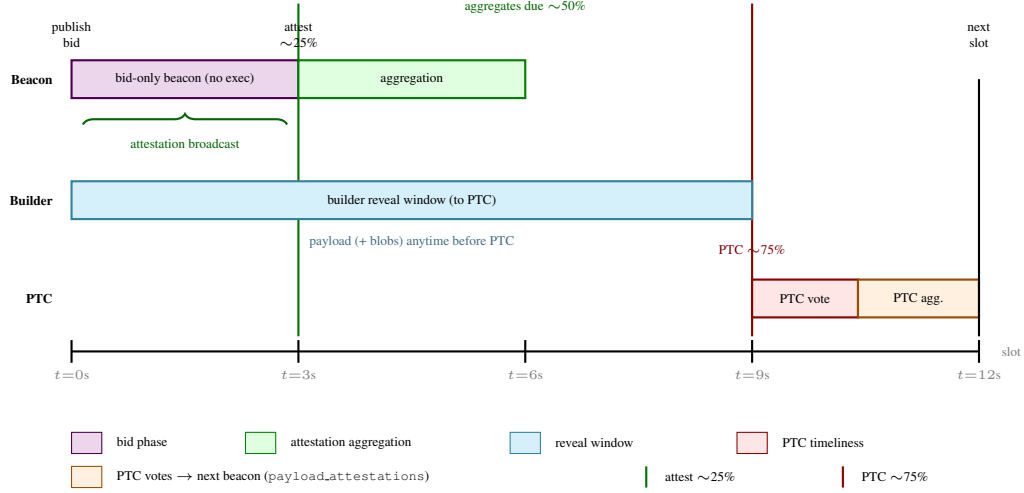


Figure 3: Overall ePBS consensus timeline across a 12s slot: bid publication, attestations at $\sim 25\%$ followed by aggregation (due by $\sim 50\%$), builder reveal window, PTC timeliness votes near $\sim 75\%$, and carry-over of aggregated payload attestations into the next beacon block [1, 5, 6, 15]. Exact constants follow the live Gloas specs.

EMPTY

The beacon block is included but the payload is not revealed in time. If the recorded obligation reaches the same-slot attestation quorum, the proposer is still paid and the builder loses the bid amount (Path B). Below the quorum, the obligation is dropped and the builder is not charged (P5) [6].

SKIPPED

There is no beacon block, for example because the proposer is offline. This is the same as a missed slot today [1].

PENDING

The node’s local view has not yet resolved whether the payload is available, either because the PTC has not voted or because the node lacks the payload [6].

5.4 Worked phase walkthrough

The ePBS security analysis divides the slot into phases [6]. Fig. 5 shows what each actor does in each phase when everything goes right; the paragraphs below also cover the branches where no block is published, a bid is invalid, or the builder withholds.

Phase 0: before the slot. Builders register and stake. The proposer can broadcast its preferences (fee recipient and target gas limit) as a `SignedProposerPreferences` message, and builders construct bids against them. Each node’s local view carries the payload status of earlier slots [5, 6].

Phase 1: slot start. The proposer selects a bid and publishes the beacon block containing the winning `SignedExecutionPayloadBid`; processing that block records the builder’s payment obligation (Sec. 6) [6]. If no beacon block is published, the slot is **SKIPPED**: there is no bid and no payment obligation [6].

Phase 1b: block receipt and local validation of the bid. Builders, attesters, and PTC members store the block, and attesters process it and update their fork-choice store. A block whose bid fails validation, for example because the builder cannot cover the bid or the bid names the wrong parent, is rejected and never admitted [5, 6].

Phase 2: attestation deadline ($\sim 25\%$). Attesters vote on the beacon block without executing the payload, and the votes are aggregated. The builder watches these attestations to decide when to release the payload; whether or not the payload has arrived does not change what attesters do at this point [6].

Phase 3: builder reveal window. The builder broadcasts the payload envelope and its blob data. The proposer, attesters, and PTC members receive them, and attesters run the execution and data-availability checks. Whether a node can treat the branch as **FULL** depends on what its local store has received [6]. If the builder withholds, either to exercise the free option or because the beacon block was not timely, no payload arrives and the slot heads toward **EMPTY** [1, 6].

Phase 4: PTC deadline ($\sim 75\%$). PTC members report whether the promised payload arrived on time and whether its blob data was available in their view. They do not check that the payload executes correctly [1, 6, 15]. Their votes are aggregated and included in the beacon block of slot $N+1$ as `payload_attestations`, where fork choice resolves the slot [5, 6].

6 The payment mechanism

The proposer commits to a builder’s bid before it sees the payload, so the protocol itself has to guarantee the payment; under MEV-Boost the relay’s escrow did that job [1, 16]. ePBS does it in three stages: an IOU is recorded when the bid is included, the IOU is *settled* (turned into a queued withdrawal) by one of three routes, and a later block executes the withdrawal. Paths A, B, and C are the security analysis’s names for the three settlement routes; the specification does not name them [5, 6]. Fig. 6 traces the routes and Table 5 lists the outcomes. The mechanism covers only the trustless payment field, `bid.value`. A bid can also advertise an `execution_payment`, which the protocol never reads: no IOU is recorded for it, and the proposer has to trust the builder to pay it [6].

Stage 1: an IOU, not a transfer. When the beacon block for slot N (in epoch e) is processed, `process_execution_payload_bid` checks that the builder can cover `bid.value` and writes a `BuilderPendingPayment` into the state. It holds the amount owed to the proposer’s fee recipient and a *weight* counter that starts at zero. No balance moves yet. IOUs are kept in a ring buffer with one entry per slot of the current and previous epoch, so an IOU written in epoch e stays there until the end of epoch $e+1$. No IOU is recorded when `bid.value = 0`, which covers every self-build [5].

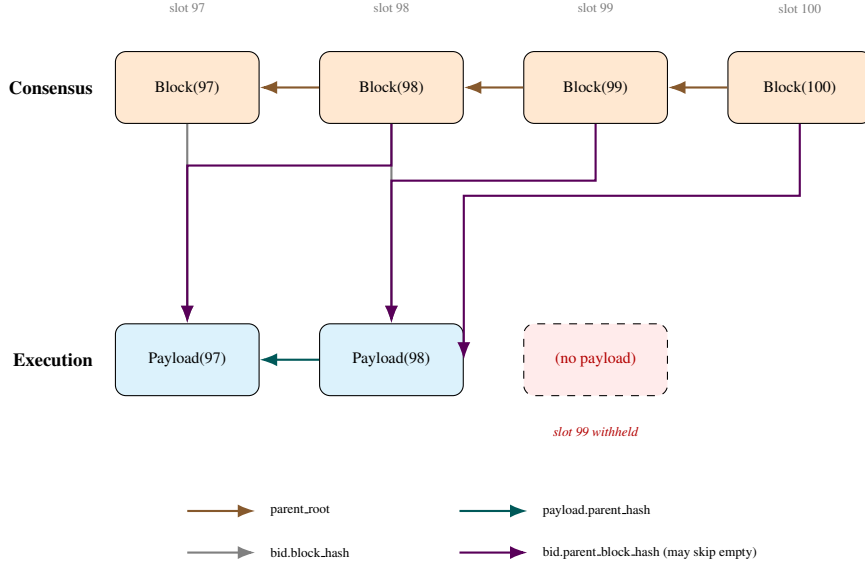


Figure 4: The two-chain model: consensus blocks commit to bids and payload hashes, and execution payloads extend the execution chain when they are revealed [6].

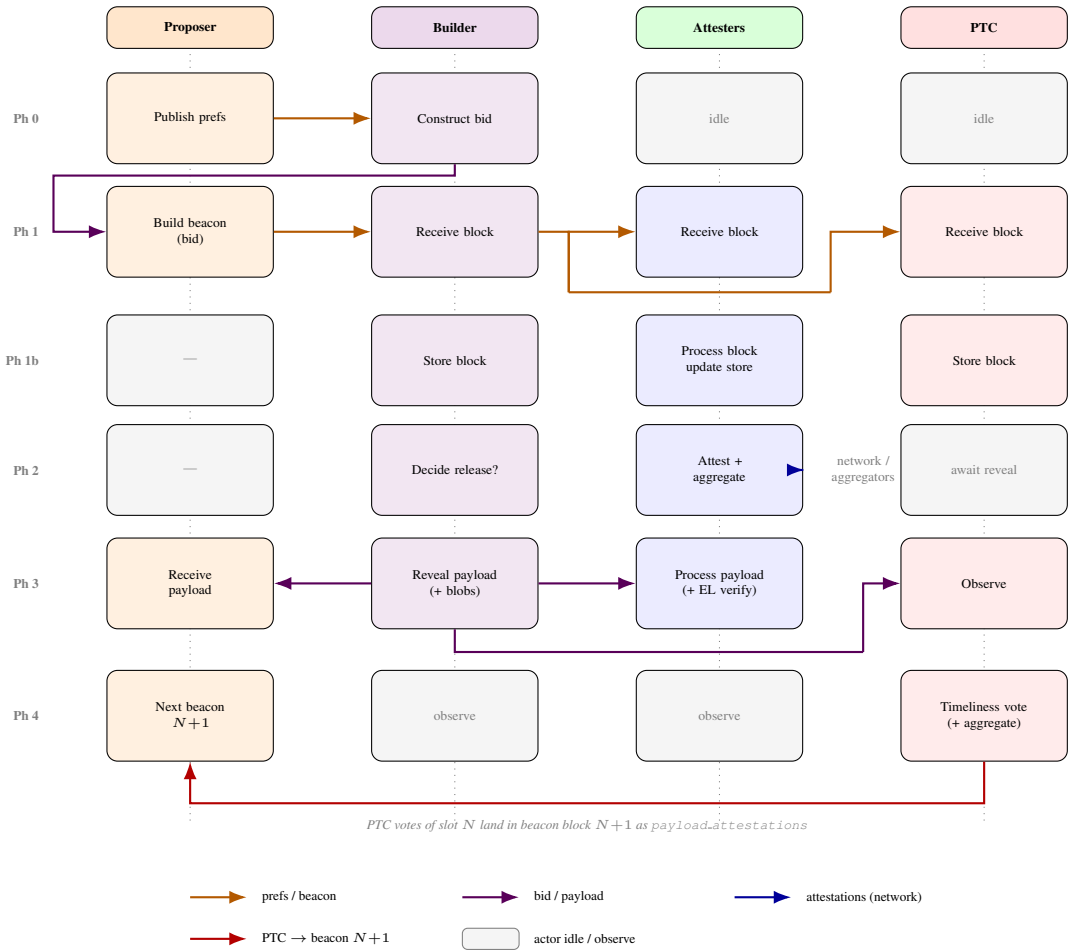


Figure 5: What each actor does in each phase of slot N when everything goes right, from the proposer's preferences (Phase 0) to the PTC votes that land in block $N+1$ (Phase 4) [6].

Path A: a successor builds on the payload. If the builder reveals and the next canonical block declares slot N FULL (its bid names slot N 's payload as parent), that block's `process_parent_execution_payload` calls `settle_builder_payment`. The IOU moves into the withdrawal queue and its buffer entry is cleared. This is the normal case and usually happens at slot $N+1$. Clearing the entry is what stops Path B from paying the same bid a second time [5, 6].

Path B: the slot stays empty. If slot N ends up EMPTY, because the builder withheld or the payload was not accepted in time, the IOU stays in the buffer. Every same-slot attestation for slot N 's beacon block that is included on chain adds the attester's effective balance to the IOU's weight, once per validator; PTC votes do not count. At the end of epoch $e+1$, `process_builder_pending_payments` checks the IOU once. If the weight is at least 60% of the stake assigned to one slot, the IOU is queued and the proposer is paid although no payload appeared. Otherwise the IOU is dropped and the builder pays nothing [5, 6]. The quorum is what lets both parties rely on the rule. A timely beacon block collects enough honest same-slot votes to reach it, so its proposer is paid whatever the builder does (property P6). A builder that withheld because the beacon block was late or contested is not charged, because such a block does not reach the quorum (property P5). The security analysis derives the 60% as 40% plus 20%: an honest builder reveals only after it has seen at least 40% of the attestation weight, and Byzantine stake is assumed to stay below 20% [6].

Path C: a late successor after missed slots. Path A needs the IOU to still be in the buffer. Suppose the builder revealed, but every following slot was missed until epoch $e+2$ or later. By the time a canonical block declares slot N FULL, the IOU has been dropped, at zero weight because no block was there to include attestations. `apply_parent_execution_payload` then queues the withdrawal directly from the cached bid, so a revealed payload still pays its proposer. Path C is a rare corner case [5, 6].

Equivocation. If the slot- N proposer is slashed for proposing two blocks while the IOU is still in the buffer, `process_proposer_slashing` clears the IOU and no path pays for that bid [5].

Stage 3: from the queue to the execution layer. Every path ends with a `BuilderPendingWithdrawal` in the withdrawal queue. The builder's balance is debited, and the proposer's fee recipient credited on the execution layer, only when a later block whose parent is FULL processes the queue as part of its withdrawals; a block whose parent is EMPTY skips withdrawals. The promise that the proposer is "eventually credited" therefore assumes that FULL blocks keep appearing, which the security analysis states as a liveness assumption [5, 6].

7 Fork choice

Fork choice must track pairs of block root and payload status rather than beacon roots alone, so that FULL and EMPTY children of the same block compete correctly [1, 5, 6]. Phase 5, at the start of the next slot, resolves any remaining PENDING views using PTC votes and the payloads each node has stored [6].

Head selection. The security analysis documents head selection as an ordering on several keys, with lenient tie-breakers and stricter rules for block construction [6]. Admission checks and limits in both directions constrain when payload information can shift fork-choice weight [6]. Gloas removes or redefines some pre-ePBS proposer-boost rules; the specs and the security analysis are more reliable on these details than secondary blog posts [5, 6].

Why the change was necessary. If fork choice tracked beacon roots alone, attestation weight could not distinguish a withheld payload from a revealed one. The EIP enforces builder reveal safety and builder withhold safety by letting attestations signal whether a slot is full, empty, or skipped [1, 6].

8 Security properties and attack surface

8.1 Externally observable properties

The ePBS security analysis sorts its guarantees (P1–P6 and P_{valid}) into three groups: properties that hold whether or not payment is trustless,

properties whose assumptions differ between trustless and non-trustless payment, and a property defined only for trustless payment [6]. Informally, and using the analysis's own names, P1 is block safety; P2 states that the payload execution deadline is the start of the next slot; P3 is data availability for chain inclusion; P_{valid} is chain validity (two admissible parent hashes); P4 and P5 protect builders that reveal and builders that withhold, respectively; and P6 is unconditional proposer payment [6]. The broader goals of publication safety, proposer safety, and permissionlessness are design requirements from the earlier ePBS literature, not property labels in the analysis [8, 16]. The formal proofs (§9 of the analysis) are still in progress upstream [6].

8.2 Structural assumptions and calibration

The properties hold under structural assumptions S1–S3 and a calibration that links proposer boost, the Byzantine budget, and the payment quorum, which the analysis summarizes as $60\% = 40\% + 20\%$ [6]. Further algorithmic assumptions cover honest reorg behavior and the PTC honesty thresholds [6].

8.3 Catalogued attacks (conceptual)

ePBS removes two attack surfaces: unbundling through a relay, and the relay as a mandatory point of censorship [1, 8]. Neuder and Drake note that in a two-block design such as TBHL an attacker who controls two consecutive slots effectively controls four, which weakens HLMD-GHOST with proposer boost [8]. For EIP-7732 itself, proposers and attesters that control consecutive blocks and more than 20% of the stake can reorg a builder's payload and still charge it the bid; a 2025 SoK puts the required stake at nearly 6.7M ETH [1, 33].

9 Economic and market-structure implications

9.1 Free option

Once its bid is included, the builder holds a short-dated option: it can reveal or withhold the payload until the PTC deadline [32, 34]. Reported exercise rates are about 0.82% of blocks on average with an 8s window, 3–6% on high-volatility days, and higher for builders that focus on CEX–DEX arbitrage [32, 34]. Mazorra et al. (v1, 2025-09) report a median daily aggregate option value of about 0.84 ETH over a 14-month CEX–DEX sample (2024-01 to 2025-03, 8s window); Flashbots' Part II write-up describes the searcher sample as 23 CEX–DEX searchers. The distribution is right-skewed: Flashbots' Part II write-up of the same dataset reports an average near 3.67 ETH per day, and the value exceeds 40 ETH on the most volatile days. Flashbots' figure of about \$1.26M per day is a different quantity: a single-day scenario estimate that assumes \$1M notional traded every 8s in ETH–USDC on 2025-07-21, realizing about \$117 per interval over 10,800 intervals, rather than a market-wide aggregate cost. The methodologies are CONFLICTING, so the figures should be cited as separate measures and compared only after matching the quantity measured and the population [32, 34, 35]. Proposed mitigations include an earlier PTC deadline, penalties for not revealing (amounts of about 0.5–1 ETH or more have been discussed), and threshold encryption of the payload to the PTC. None has been adopted in the protocol [1, 5, 34, 35]. Mazorra et al. (§5; arXiv 2509.24849, v1 2025-09) evaluate several of these in their model, which assumes a dual-PTC-deadline design with an 8s option window. Shortening the window from 8s to 6s cuts the exercise probability by more than 33%, to 4s by more than 50%, and to 2s by more than 77%. A static penalty of 0.075 ETH reduces it by about 75%, and 0.15 ETH by about 83%. A dynamic penalty set by online gradient descent with a 0.1% target achieves 0.096% at an average penalty of about 0.104 ETH. These results are conditional on the model's assumptions [32].

9.2 Timing-game relocation

Under ePBS the beacon proposer can still delay its block toward the attestation deadline, as under MEV-Boost, and the builder gains a second timing game: it can delay its payload reveal up to the PTC deadline [36, 37].

9.3 Concentration

The builder oligopoly is attributed mainly to exclusive private order flow rather than to latency alone [38, 39]. Wu et al. report that two builders, beaverbuild and Titan, won more than 90% of auctions as of April 2025 [38]. In the block sample of Mazorra et al. (blocks with CEX–DEX trades, 2024-01 to 2025-03), beaverbuild built about 49% and Titan about 29% of

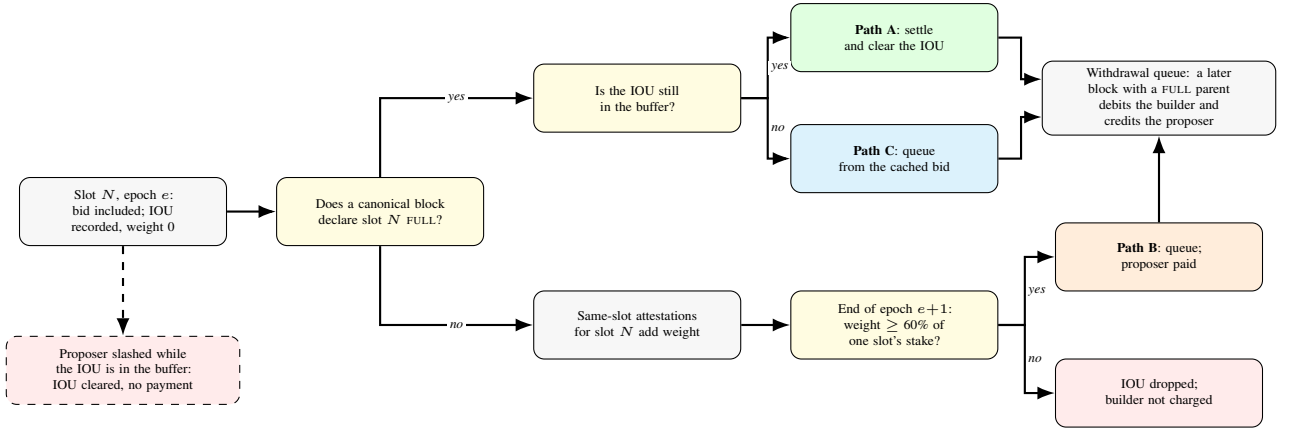


Figure 6: What happens to the IOU of a bid with $\text{value} > 0$. It is settled by Path A, B, or C, or dropped; a settled IOU becomes a withdrawal that a later block executes [5, 6].

Table 5: Outcomes for a bid with $\text{value} > 0$, by settlement path [5, 6].

Case	Trigger and timing	Proposer	Builder
Path A	A canonical block declares slot N FULL while the IOU is in the buffer; usually at slot $N+1$	Paid	Pays; its payload is canonical
Path B	Slot N stays EMPTY, and at the end of epoch $e+1$ the weight is at least 60% of one slot's stake	Paid	Pays, with no payload that slot
Below quorum	Slot N stays EMPTY, and the weight is below 60%	Not paid	Not charged (P5)
Path C	A canonical block in epoch $e+2$ or later declares slot N FULL; every slot in between was missed	Paid	Pays; its payload is canonical
Equivocation	The slot- N proposer is slashed while the IOU is in the buffer	Not paid	Not charged
Bid not included	The bid never reaches a canonical beacon block	Not paid	Not charged [16]

Table 6: Attack classes and the mechanisms that address them (conceptual) [1, 6].

Class	Idea	Mitigation
Proposer equivocation and unbundling	Conflicting beacon blocks split the attesters	Headlock-style local builder selection; conditional payment [12]
PTC split view	A builder that controls a network partition colludes with a single malicious PTC member to split views on payload presence	Not penalized; the EIP names a 2/3 PTC threshold or new slashing conditions as possible fixes [1]
Post-state monopoly	A late reveal keeps exclusive knowledge of the resulting state	In tension with a late PTC deadline and pipelining [31]
Free-option withholding	Rational refusal to reveal after unfavorable price moves	Economic mitigations (Sec. 9); not a safety failure in the classical sense [1, 32]

blocks [32]. ePBS removes unfunded relays as mandatory brokers, but it raises the capital and operational barriers for builders, who must stake and bid in a more sophisticated way. Concentration may therefore move rather than decline [1, 8, 40]. Titan Builder expects the auction to shift from the open second-price dynamics of MEV-Boost relays to a first-price blind auction under ePBS [40]. Flashbots likewise expects in-protocol trustless payments to lock in a sealed-bid, first-price block auction [49].

9.4 Censorship

ePBS removes relays as a censorship point, but builders can still exclude or delay transactions, so ePBS alone does not deliver censorship resistance [41]. EIP-7732 is designed to be compatible with FOCIL (EIP-7805), a fork-choice enforced inclusion-list design [1, 42]. When ePBS was chosen as Glamsterdam's consensus-layer headliner, FOCIL was left Considered for Inclusion [3, 4]. It is not in Glamsterdam's final scope and is now Scheduled for Inclusion in the next upgrade, Hegóta [2, 43].

9.5 Validator economics

Validators get a payment that no longer depends on the builder revealing [1]. The effect on revenue is less clear: Titan Builder calls the net effect of the auction change unclear [40], while potuz argues that validators gain either execution-gas revenue from higher throughput or lower hardware requirements [24]. Titan Builder also expects the Builder API to support multiple builders directly, so validators would no longer need the mev-boost sidecar to multiplex Builder API calls [40].

9.6 Fork-choice resolution at slot $N+1$

Phase 5 begins at the start of the next slot. Each client combines the PTC votes, the payloads it has stored, and the attestation weight to choose a head that accounts for payload status [6]. A node that still lacks a payload may keep it PENDING until the payload arrives or the empty branch wins [6].

Why fork choice uses pairs of root and status. If FULL and EMPTY were treated as the same beacon root, fork choice would lose the economic and availability distinction that the PTC exists to report [1, 6]. Fork-choice nodes therefore carry payload status, and head selection orders nodes by weight, then status preference, then tie-breakers, as documented in the security analysis [6].

Admission and authority limits. Admission checks decide when a newly observed payload may upgrade a branch's status, and limits in both directions prevent payload information from arbitrarily rewriting earlier consensus weight [6]. For the proposer-boost rules that Gloas removes or redefines, consensus-specs is the authoritative source [5, 6].

10 Additional protocol mechanics

10.1 Conceptual history

The idea developed from Buterin's two-slot PBS sketch [23], through MEV-Boost at the Merge [7], Neuder and Drake's argument that ePBS is viable [8], the PTC design [15], and potuz's design constraints [16], to the creation of EIP-7732 in June 2024 [1]. ACDC #162 and ACDE #218 confirmed ePBS as Glamsterdam's main consensus-layer change, and the consensus work lives in the Gloas branch of consensus-specs [3–5]. Potuz

describes the relay as “the current bug” and the EIP as solving it “in a minimal viable way”, granting that slot auctions or execution tickets “may be better” while admitting not knowing how to implement them [26].

10.2 Two-phase anatomy and PTC in detail

In the first phase, the beacon block carries the `SignedExecutionPayloadBid`, which contains the block hash, value, builder index, slot, blob KZG commitments, and execution-requests root. The beacon block no longer embeds the execution payload [1, 5]. In the second phase, the builder publishes the `SignedExecutionPayloadEnvelope`, which contains the full payload, the execution requests, the builder index, and the beacon-block and parent-beacon-block roots. The blob KZG commitments are already committed in the bid [5]. Attesters vote at about 25% of the slot on the consensus transition only. PTC members vote at about 75% on whether the payload is present and blob data is available; they do not vote on execution validity [1, 5]. The PTC has 512 seats per slot, filled from that slot’s beacon committees by balance-weighted sampling, so one validator can hold more than one seat; the committee can be computed up to one epoch ahead. Votes from slot N are aggregated into beacon blocks at slot $N+1$, which carry at most four payload attestations [1, 5]. An alternative PTC design splits the vote into two deadlines, an earlier one for the payload and a later one for blobs, to leave time for both execution and blob propagation [32, 44]. Flashbots’ free-option analysis assumes this design, with the payload deadline at about 4s and the blob deadline at about 10s, which gives the builder an option window of about 8s; the pinned Gloas spec instead has a single PTC deadline at 75% of the slot [5, 34].

The field lists below follow consensus-specs at commit `ec1c01f5`, the same commit the ePBS security analysis uses [5, 6]. The live EIP-7732 page shows an older summary, with `slot` and `state_root` in the envelope and no `execution_requests_root` in the bid. The two are different SSZ schemas, and neither is a superset of the other [1].

```
BeaconBlockBody {
  ..., signed_execution_payload_bid,
  payload_attestations, parent_execution_requests
}
ExecutionPayloadBid {
  block_hash, value, builder_index, slot,
  blob_kzg_commitments,
  execution_requests_root, ...
}
ExecutionPayloadEnvelope {
  payload, execution_requests, builder_index,
  beacon_block_root, parent_beacon_block_root
}
PayloadAttestationData {
  beacon_block_root, slot,
  payload_present, blob_data_available
}
```

The beacon state drops the latest execution payload header and adds the builder registry, a payload-availability bitvector, and payment queues [1, 5].

10.3 Builder registry and gossip

Builders register with their own withdrawal-credential rules, a minimum stake commonly cited as 1 ETH, and withdrawal delays of tens of epochs [1, 5]. Builders do not attest or propose beacon blocks; their stake is collateral for trustless payments [1, 5]. At the pinned commit, three gossip topics carry the fair exchange: `execution_payload_bid` for builder bids (named `execution_payload_header` in the historical validator spec), `execution_payload` for the full envelope, and `payload_attestation_message` for PTC votes. Under ePBS, publishing blob sidecars becomes the builder’s responsibility [5, 27]. Two new signature domains are introduced: `DOMAIN_BEACON_BUILDER` (`0x0B000000` at the pinned commit; `0x1B000000` only in the pre-Gloas feature spec) and `DOMAIN_PTC_ATTESTER` (`0x0C000000`) [1, 5, 27].

10.4 Deferral and scaling claim

Removing execution from the attestation critical path allows the longer windows for execution and blob broadcast that are commonly cited; arguments for inclusion use illustrative figures of about 2s growing to about 8–9s [24]. Validation work moves to the next proposer and to other validators later in the slot [1].

10.5 The Builder API that ePBS replaces

MEV-Boost’s Builder API (status, validator registration, `getHeader`, and blinded block submission) is the HTTP interface for the fair exchange that ePBS aims to remove from the critical path [1, 20]. `getHeader` is where

trust is concentrated: without the relay, the consensus-layer client cannot cryptographically verify the value a builder claims [7].

10.6 Honest withholding when the beacon block is not timely

The specification distinguishes strategic withholding to exercise the free option from honest withholding, where the beacon block containing the bid does not become the timely canonical head. Honest withholding is treated as if no block had been produced, not as a slashable offense [5, 6].

11 Design alternatives and debate

PEPC. Protocol-enforced proposer commitments generalize the single enshrined auction into trustless infrastructure for proposer commitments in general. The original PEPC proposal (October 2022) predates EIP-7732 and presents a general commitment framework that does not depend on it [45]. A core developer has described other allocation designs, such as slot auctions and execution tickets, as iterations that are simpler to specify once ePBS is in place [26]; by analogy, our assessment is that PEPC would build on ePBS rather than replace it.

Execution tickets. Execution tickets shield validators from MEV pressure by selling the right to propose execution payloads in a separate market. The proposal acknowledges risks from secondary markets spanning multiple slots and from cartelization [46].

Delayed execution (EIP-7886). EIP-7886 is claimed to deliver much of the pipelining benefit without enshrining PBS. It avoids the complexity of payload separation, unconditional payment, and the fork-choice changes, and it has no post-state monopoly window, but it does not remove the need to trust relays [31, 47].

FOCIL. Inclusion lists address censorship by builders, which remains after relays are removed. Most roadmaps place FOCIL after Glamsterdam [42, 43].

Sigma Prime critique. Sigma Prime argues that MEV-Boost “has been performing well, with infrequent incidents that have had little to no impact”, that the main scaling gains (payload split and delayed execution) can be delivered without trustless payments, that the staked-builder requirement “adds friction to new players for an already highly centralized market”, and that committing early to one flavor of ePBS gives up flexibility while slot auctions, execution-layer payment paths, and MEV-burn integration are still being researched [25]. Core developers nevertheless confirmed EIP-7732 as Glamsterdam’s consensus-layer headliner; in the stakeholder feedback compiled for that decision, nine stakeholders ranked ePBS first, with strong emphasis on a trustless proposer-builder exchange [3, 48].

Open questions. Several questions remain open. Free-option mitigations have been evaluated only in models (Mazorra et al. §5), and none is in the protocol [32]; builder capital concentration has to be weighed against MEV-burn and other designs that are still being researched [25]; and the formal proofs of the security properties are still in progress [6]. In our assessment, the PTC manipulation thresholds, the combined properties of ePBS and FOCIL, and the interaction with single-slot finality (SSF) also lack published analysis.

Expanded security property map

Assumptions S1–S3. The proof sketches in the security analysis rest on structural assumptions about honest online weight, the Byzantine budget, and majority conditions for chain inclusion and the PTC [6]. Side conditions, such as alignment of payment weight and timely inclusion of slashings, appear in §9 and are part of the still-provisional proofs [6]. In particular, the trustless P5 proof relies on $I_{\text{payweight}}$ and I_{slash} , which the main statement now lists alongside S1 and S2. This was a mismatch between statement and proof, found in review and since corrected; it is not a demonstrated attack [6].

Summary of the adversarial picture. Compared with MEV-Boost, ePBS removes theft from relay escrow and relay-controlled unblinding as mandatory trust points, moves timing games to builders, and introduces new attack surfaces in the committee and the free option [1, 6, 34, 37]. The net effect on centralization is still debated. Removing relays takes a small,

Table 7: Builder API functions in MEV-Boost and their ePBS counterparts [5, 20].

Builder API function	ePBS counterpart (conceptual)
Register fee recipient and gas limit	Proposer preferences (<code>SignedProposerPreferences</code> : fee recipient, target gas limit)
getHeader blinded bid	Bid gossip topic; bid in the beacon block
Submit blinded block and unblind payload	Envelope gossip and PTC timeliness vote
Relay honesty as escrow	Consensus-layer payment, PTC, and fork choice

Table 8: Security properties, with informal labels from the security analysis; the formal proofs are still provisional [6].

ID	Intent
P1	Block safety: a valid-bid block from an honest proposer extending a forever-canonical parent stays canonical, under S1 and S2 [6].
P2	The payload execution deadline is the start of the next slot, so execution is off the attestation critical path [6].
P3	Data availability for chain inclusion: a payload hash on the canonical payload hash chain was available (payload valid, blob data available) at some honest node, under S2 [6].
P4	Builder revealing protection: if an honest builder reveals payload P in slot N , $P.block.hash$ lands on the canonical payload hash chain; in the trustless case this holds under S1, S2, S3, $L.verify$, $L.successor$, and timeliness, and in the non-trustless case through a confirmation rule instead of consensus-layer payment [6].
P5	Builder withholding protection (trustless case): a builder that withholds and does not reach the quorum is not charged. Main hypotheses: S1, S2, and the payment-weight and timely-slashing-inclusion side conditions ($I_{payweight}$, I_{slash}); the proof additionally uses A1a and the G-assumptions [6].
P6	Unconditional proposer payment (trustless case): a bid included on time eventually has its payment queued through Path A or B, under hypotheses on attestation inclusion [6].
P_{valid}	Chain validity: every canonical block’s $bid.parent.block.hash$ equals one of exactly two admissible hashes from its parent’s bid (parent FULL or EMPTY); no third value is admissible [6].

centralized set of trusted operators out of the pipeline [8], but the staked-builder requirement makes block building more capital intensive and adds friction for new entrants to an already concentrated builder market [25, 40].

Expanded economics notes

Option window. In the illustrative timeline with the PTC deadline at about 75% of the slot, the option window is about eight seconds [5, 34]. Rough estimates of the option’s value grow with price volatility over the window and with the notional value exposed. Diversified builders give up substantial non-directional MEV when they withhold, which weakens the incentive to exercise the option except for strategies that are mostly directional [34, 35].

Feedback risk. Builders that withhold can bid more aggressively because they price in the option to walk away. If validators stop blocklisting such builders for that reason, builders that always reveal can be outbid and pushed out of the market. Flashbots has flagged this self-reinforcing cycle [34].

Relay endgame. Some analyses expect relay-like services to survive in a reduced role, for example as capital providers for under-collateralized bids, for builder discovery, or for fast payload distribution, but no longer as mandatory escrow [40, 49]. ePBS ends the protocol’s reliance on MEV-Boost’s blinded HTTP fair exchange, and staking guides advise removing the MEV-Boost sidecar after the fork; relays, however, are expected to stay in use in some form [17, 49, 50].

12 Open problems, trade-offs, and roadmap

- Design choices: a block auction or a slot auction; one PTC deadline or two; the penalty regime; and whether payment is enforced by the consensus layer or the execution layer [6, 25, 32, 34].
- Free-option mitigations versus pipelining and scaling goals: Mazorra et al. evaluate shorter windows and static or dynamic penalties only in a model, and no mechanism has been adopted [32, 34].
- Completing the formal proofs [6].
- Relation to the roadmap: FOCIL [42] and MEV-burn [51] complement ePBS, and potuz calls ePBS simply compatible with FOCIL [24]; EIP-7886 [47] delivers much of the pipelining without ePBS and so partly substitutes for it [31]; execution tickets [46] are described by their authors as orthogonal to PBS; and PEPC [45] generalizes it and would build on top of it (our assessment, not a demonstrated dependency).

- MEV-Boost and relays after adoption: the protocol no longer relies on them, but relays may change their role rather than disappear at once [17, 49, 52].

13 Conclusion

EIP-7732 enshrines PBS in the protocol to remove the need to trust relays when producing Ethereum blocks. It does so with bids inside beacon blocks, a committee that attests to payload timeliness, deferred execution, and proposer payment enforced by the protocol. That payment does not depend on the builder’s behavior after committing and is settled by turning a recorded obligation into a withdrawal [1, 5, 6]. The two-chain model and the slot phases define FULL, EMPTY, and SKIPPED outcomes precisely enough for fork choice and payment [5, 6]. ePBS closes the relay equivocation and escrow-theft attacks but introduces new attack surfaces around the free option and the committee [1, 6, 34]. Economically, builder concentration and timing games move to new places rather than disappear, and censorship resistance still requires FOCIL [37, 40, 42]. ePBS is Glamsterdam’s main consensus-layer change and, in potuz’s framing, a minimal fix for relay trust; many research questions remain open [3, 17, 26].

References

- [1] F. D’Amato, N. Flaig, B. Monnot, M. Neuder, Potuz, J. Traglia, and T. Tsao, “EIP-7732: Enshrined Proposer-Builder Separation,” Ethereum Improvement Proposals, Standards Track / Core (Last Call, deadline 2026-11-01), Jun. 2024. [Online]. Available: <https://eips.ethereum.org/EIPS/eip-7732>
- [2] T. Beiko, A. Stokes, A. Dietrichs, Nix, and P. Jayanthi, “EIP-7773: Hardfork Meta - Glamsterdam,” Ethereum Improvement Proposals, Sep. 2024. [Online]. Available: <https://eips.ethereum.org/EIPS/eip-7773>
- [3] Ethereum All Core Devs, “All Core Devs - Execution (ACDE) #218, August 14, 2025,” Ethereum Magicians, Aug. 2025. [Online]. Available: <https://ethereum-magicians.org/t/all-core-devs-execution-acde-218-august-14-2025/24979>
- [4] “All Core Devs Consensus (ACDC) #162, August 7, 2025,” Ethereum Magicians, 2025. [Online]. Available: <https://ethereum-magicians.org/t/all-core-devs-consensus-acdc-162-august-07-2025/24919>
- [5] Ethereum Foundation, “consensus-specs: Gloas fork specifications (EIP-7732),” GitHub, commit `ec1c01f5`, 2026. [Online]. Available:

- <https://github.com/ethereum/consensus-specs/tree/ec1c01f5f10fb60636a022ae8944c912b6da35f8/specs/gloas>
- [6] ePBS Security Analysis Contributors, “ePBS: From Spec to Security Properties,” GitHub, commit 5feba76f, Jun. 2026. [Online]. Available: <https://github.com/ethereum/epbs-security-analysis/tree/5feba76f2c1fcc33789a2dbe7150e53859cfe0e5>
 - [7] Flashbots, “MEV-Boost,” Flashbots Docs (Introduction, Relay, and Risks pages), accessed Oct. 2026. [Online]. Available: <https://docs.flashbots.net/flashbots-mev-boost/introduction>
 - [8] M. Neuder and J. Drake, “Why enshrine Proposer-Builder Separation? A viable path to ePBS,” ethresear.ch, May 2023. [Online]. Available: <https://ethresear.ch/t/why-enshrine-proposer-builder-separation-a-viable-path-to-epbs/15710>
 - [9] T. Wahrstätter, “MEV-Boost dashboard,” mevboost.pics. [Online]. Available: <https://mevboost.pics>
 - [10] M. Nijkerk, “Blocknative to suspend MEV-Boost relay after economics fail to ‘materialize,’” CoinDesk, Sep. 26, 2023. [Online]. Available: <https://www.coindesk.com/tech/2023/09/26/blocknative-to-suspend-mev-boost-relay-after-economics-fail-to-materialize>
 - [11] Flashbots, “Post mortem: April 3rd, 2023 mev-boost relay incident and related timing issue,” Flashbots Collective, Apr. 2023. [Online]. Available: <https://collective.flashbots.net/t/post-mortem-april-3rd-2023-mev-boost-relay-incident-and-related-timing-issue/1540>
 - [12] F. D’Amato and M. Neuder, “Equivocation attacks in mev-boost and ePBS,” ethresear.ch, Apr. 2023. [Online]. Available: <https://ethresear.ch/t/equivocation-attacks-in-mev-boost-and-epbs/15338>
 - [13] T. Sargent, “Corrupt commitments: Proposer equivocation bug in Helix MEV relay,” Asymmetric Research Blog, Dec. 2024. [Online]. Available: <https://blog.asymmetric.re/corrupt-commitments-proposer-equivocation-bug-in-helix-mev-relay/>
 - [14] Labrys, “MEV Watch,” mevwatch.info. [Online]. Available: <https://www.mevwatch.info>
 - [15] M. Neuder and F. D’Amato, “Payload-timeliness committee (PTC) — an ePBS design,” ethresear.ch, Jul. 2023. [Online]. Available: <https://ethresear.ch/t/payload-timeliness-committee-ptc-an-epbs-design/16054>
 - [16] Potuz, “ePBS design constraints,” ethresear.ch, Feb. 2024. [Online]. Available: <https://ethresear.ch/t/epbs-design-constraints/18728>
 - [17] ethereum.org contributors, “Glamsterdam,” ethereum.org Roadmap, updated Sep. 2026. [Online]. Available: <https://ethereum.org/roadmap/glamsterdam/>
 - [18] ethereum.org contributors, “Proposer-builder separation,” ethereum.org Roadmap. [Online]. Available: <https://ethereum.org/roadmap/pbs/>
 - [19] Flashbots, “MEV-Boost Relay architecture,” GitHub, 2023. [Online]. Available: <https://github.com/flashbots/mev-boost-relay/blob/main/ARCHITECTURE.md>
 - [20] Ethereum, “builder-specs: Specification for the external block builders,” GitHub repository. [Online]. Available: <https://github.com/ethereum/builder-specs>
 - [21] 0x_emperor, “Ethereum’s supply chain, Part 1: PoW, PoS, PBS,” Umbra Research, Sep. 2023. [Online]. Available: <https://www.umbraresearch.xyz/writings/ethereum-supply-chain-part-1>
 - [22] B. Lindrea, “OFAC-compliant blocks on Ethereum hit three-month low of 47%,” Cointelegraph, Feb. 14, 2023. [Online]. Available: <https://cointelegraph.com/news/ofac-compliant-blocks-on-ethereum-hits-three-month-low-of-47>
 - [23] V. Buterin, “Two-slot proposer/builder separation,” ethresear.ch, 2021. [Online]. Available: <https://ethresear.ch/t/two-slot-proposer-builder-separation/10980>
 - [24] Potuz, “EIP-7732 the case for inclusion in Glamsterdam,” Ethereum Magicians, May 2025. [Online]. Available: <https://ethereum-magicians.org/t/eip-7732-the-case-for-inclusion-in-glamsterdam/24306>
 - [25] Dapplion, “OPINION: The case against EIP-7732 for Glamsterdam,” Sigma Prime Blog, Jul. 2025. [Online]. Available: <https://blog.sigmaprime.io/glamsterdam-eip7732.html>
 - [26] Potuz, “Reply #3 in the EIP-7732: Enshrined Proposer-Builder Separation (ePBS) discussion thread,” Ethereum Magicians, Apr. 13, 2024. [Online]. Available: <https://ethereum-magicians.org/t/eip-7732-enshrined-proposer-builder-separation-epbs/19634/3>
 - [27] Ethereum Foundation, “consensus-specs: EIP-7732 feature specifications (historical),” GitHub, commit 9e2dc0bc, Aug. 2025. [Online]. Available: https://github.com/ethereum/consensus-specs/tree/9e2dc0bcf9aa17b549e1df4712b16a5709d74119/specs/_features/eip7732
 - [28] “Staked Builder API for Glamsterdam,” ethereum/builder-specs pull request #138, Jun. 2026. [Online]. Available: <https://github.com/ethereum/builder-specs/pull/138>
 - [29] Ethereum Foundation, “consensus-specs: EIP-7732 feature specification (_features/eip7732),” GitHub, release v1.5.0-alpha.10, Dec. 2024. [Online]. Available: https://github.com/ethereum/consensus-specs/tree/v1.5.0-alpha.10/specs/_features/eip7732
 - [30] Ethereum Foundation, “consensus-specs: Phase 0 honest validator guide,” GitHub, commit ec1c01f5, 2026. [Online]. Available: <https://github.com/ethereum/consensus-specs/blob/ec1c01f5f10fb60636a022ae8944c912b6da35f8/specs/phase0/validator.md>
 - [31] T. Wahrstätter, “Slot restructuring: Design considerations and trade-offs,” ethresear.ch, Jun. 2025. [Online]. Available: <https://ethresear.ch/t/slot-restructuring-design-considerations-and-trade-offs/22687>
 - [32] B. Mazorra, B. Öz, C. Schlegel, and F. Wu, “The free option problem of ePBS,” arXiv:2509.24849v1, Sep. 2025. [Online]. Available: <https://arxiv.org/abs/2509.24849v1>
 - [33] M. Koegler, “SoK: Current state of Ethereum’s enshrined proposer builder separation,” arXiv:2506.18189, Jun. 2025. [Online]. Available: <https://arxiv.org/abs/2506.18189>
 - [34] C. Schlegel, 0xSybil, and B. Mazorra, “The free option problem in ePBS,” Flashbots Collective, Jul. 2025. [Online]. Available: <https://collective.flashbots.net/t/the-free-option-problem-in-epbs/5115>
 - [35] C. Schlegel, F. Wu, 0xSybil, B. Öz, and B. Mazorra, “The free option problem in ePBS, Part II,” Flashbots Collective, Aug. 2025. [Online]. Available: <https://collective.flashbots.net/t/the-free-option-problem-in-epbs-part-ii/5145>
 - [36] C. Schwarz-Schilling and M. Neuder, “Timing games: Implications and possible mitigations,” ethresear.ch, Dec. 2023. [Online]. Available: <https://ethresear.ch/t/timing-games-implications-and-possible-mitigations/17612>
 - [37] T. Tsao, “Builder reveal timing game in ePBS,” ethresear.ch, Apr. 2024. [Online]. Available: <https://ethresear.ch/t/builder-reveal-timing-game-in-epbs/19424>

- [38] F. Wu, T. Thiery, S. Leonardos, and C. Ventre, “From competition to centralization: The oligopoly in Ethereum block building auctions,” arXiv:2412.18074, Dec. 2024. [Online]. Available: <https://arxiv.org/abs/2412.18074>
- [39] S. Yang, K. Nayak, and F. Zhang, “Decentralization of Ethereum’s builder market,” arXiv:2405.01329, May 2024. [Online]. Available: <https://arxiv.org/abs/2405.01329>
- [40] Titan Builder, “Builders and relays in ePBS,” Titan Builder (Substack), Jul. 2025. [Online]. Available: <https://titanbuilder.substack.com/p/builders-and-relays-in-epbs>
- [41] EtherWorld, “Hegóta should complete the holy trinity of censorship resistance,” EtherWorld, Jan. 2026. [Online]. Available: <https://etherworld.co/hegota-should-complete-the-holy-trinity-of-censorship-resistance/>
- [42] T. Thiery, F. D’Amato, J. Ma, B. Monnot, T. Tsao, J. Kaufmann, and J. Song, “EIP-7805: Fork-choice enforced inclusion lists (FOCIL),” Ethereum Improvement Proposals (Draft), Nov. 2024. [Online]. Available: <https://eips.ethereum.org/EIPS/eip-7805>
- [43] T. Beiko, A. Stokes, A. Dietrichs, Nix, and P. Jayanthi, “EIP-8081: Hardfork Meta - Hegóta,” Ethereum Improvement Proposals, Draft, 2025. [Online]. Available: <https://eips.ethereum.org/EIPS/eip-8081>
- [44] A. Elowsson, “Dual-deadline PTC vote in ePBS,” notes.ethereum.org, Jun. 2025. [Online]. Available: <https://notes.ethereum.org/@anderselowsson/Dual-deadlinePTCvote>
- [45] B. Monnot, “Unbundling PBS: Towards protocol-enforced proposer commitments (PEPC),” ethresear.ch, Oct. 2022. [Online]. Available: <https://ethresear.ch/t/unbundling-pbs-towards-protocol-enforced-proposer-commitments-pepc/13879>
- [46] J. Drake and M. Neuder, “Execution tickets,” ethresear.ch, Dec. 2023. [Online]. Available: <https://ethresear.ch/t/execution-tickets/17944>
- [47] F. D’Amato and T. Wahrstätter, “EIP-7886: Delayed execution,” Ethereum Improvement Proposals (Stagnant), Feb. 2025. [Online]. Available: <https://eips.ethereum.org/EIPS/eip-7886>
- [48] Y. K. Chaturvedi, “EIP-7732 (ePBS) selected as Glamsterdam headliner,” EtherWorld, Aug. 2025. [Online]. Available: <https://etherworld.co/2025/08/11/eip-7732-epbs-selected-as-glamsterdam-headliner/>
- [49] C. Schlegel, 0xSybil, and M. Pan, “Trustless payments and relays,” Flashbots Collective, Jun. 2026. [Online]. Available: <https://collective.flashbots.net/t/trustless-payments-and-relays/5761>
- [50] EthStaker, “Glamsterdam features,” EthStaker Docs, 2026. [Online]. Available: <https://docs.ethstaker.org/upgrades/glamsterdam-features/>
- [51] J. Drake, “MEV burn—a simple design,” ethresear.ch, May 2023. [Online]. Available: <https://ethresear.ch/t/mev-burn-a-simple-design/15590>
- [52] M. Neuder, “Relays in a post-ePBS world,” ethresear.ch, Aug. 2023. [Online]. Available: <https://ethresear.ch/t/relays-in-a-post-epbs-world/16278>