Atho SegWit Sizing and Throughput Reference

Date: 2026-04-10

This document describes the active Atho sizing model for transaction weight, virtual size (vsize), and throughput estimation. It is intended for node operators, wallet/API integrators, and reviewers validating fee/tps assumptions.

Active Sizing Limits

From Src/Utility/const.py:

- MAX_BLOCK_SIZE_BYTES = 3,500,000
- MAX_BLOCK_WEIGHT = 14,000,000
- MAX_TRANSACTION_SIZE_BYTES = 250,000 (policy vsize cap)
- WITNESS_SCALE_FACTOR = 4

These values establish the hard envelope for mempool admission and block packing. Any throughput estimate that ignores these limits is not aligned with current consensus policy.

Canonical Metric Definitions

Atho uses SegWit-style accounting:

- weight = base_bytes * 3 + total_bytes
- vsize = ceil(weight / 4)

Interpretation:

- base bytes are weighted heavier,
- witness bytes are discounted relative to base,
- fee policy and packing policy are both driven by vsize.

Current fee baseline tied to vsize:

- fee floor 500 atoms/vB,
- minimum fee 200,000 atoms.

Base vs Witness Counting Rules

Base view (no-witness form):

- remove top-level witness fields,
- blank script_sig in input entries for deterministic no-witness sizing.

Total view:

- includes witness payload and all fields.

Implementation references:

- Src/SigWit/sigwit.py::SegWit.sizes_bytes
- binary tx serialization paths when available (ATX2 transport)

The important point is deterministic counting, not ad-hoc JSON length heuristics.

Throughput Formula

Let:

- Bv = 3,500,000 effective vbytes per block,
- T = 120 seconds per block target,
- F = packing factor (1.0 ideal, 0.95 conservative),
- Savg = average non-coinbase tx vsize.

Then:
- TPS ~= (Bv * F) / (T * Savg)

Use this as a planning model, not a guarantee. Real block composition, propagation, and tx mix always influence observed throughput.
- Current Reference vsize Profiles

Representative estimates (compressed witness, no metadata-heavy edge cases):
- 1 in / 1 out: ~513 vB
- 1 in / 2 out: ~566 vB
- 2 in / 2 out: ~615 vB
- 3 in / 2 out: ~664 vB
- 4 in / 2 out: ~713 vB

These are useful baseline points for fee estimation and capacity planning. Production applications should still measure real rolling averages from their own observed workload.
- Example TPS Bands

With T = 120:
- at 566 vB: 51.5 TPS ideal, 49.0 TPS at F=0.95
- at 615 vB: 47.4 TPS ideal, 45.1 TPS at F=0.95
- at 664 vB: 43.9 TPS ideal, 41.7 TPS at F=0.95
- at 713 vB: 40.9 TPS ideal, 38.8 TPS at F=0.95

Private flows are intentionally larger than public-only flows and therefore reduce effective T for privacy-heavy block compositions.
- Why vsize Discipline Matters

Sizing discipline directly affects:
- fair fee market behavior,
- predictable miner packing,
- mempool anti-spam controls,
- reliable cost estimation for wallets and API clients.

If two components disagree on vsize, they may disagree on minimum fee validity or replacement eligibility. That creates avoidable UX and propagation failures.
- Operational Validation Workflow

When updating tx serialization or witness rules:
- recalculate vsize metrics from live serializer output,
- run mempool admission tests for floor and minimum fee boundaries,
- compare block packing behavior under mixed tx shapes,
- verify explorer/API displayed vsize matches internal policy computation.

Related docs:
- Docs/Tx.md
- Docs/Consensus.md
- Docs/Network_Stack.md
- Policy Snapshot Coupled to Sizing

Sizing assumptions in this doc align with active policy:

- block target 120s,
- retarget 180 blocks,
- tx confirmations 10, private tx confirmations 10,
- coinbase maturity 150 blocks,
- fee floor 500 atoms/vB.

These values are not part of vsize math directly, but they define transaction lifecycle and fe
policy behavior around it.
- Practical Guidance

For production planning:
- maintain rolling averages for Savg by tx class,
- separate public-heavy and private-heavy throughput estimates,
- track packing factor (F) from observed blocks,
- keep user-facing fee estimates tied to current floor and min-fee constants.

Treat the formulas here as deterministic policy scaffolding. Then refine with real chain
telemetry from your own operating environment.