AIONET Testnet – Live Observation
BEMBH-8192 (8192-Bit Memory-Bound Hash)

Sam Nguyen-Sop
aionetprotocol@gmail.com

December 15, 2025

Abstract

This document presents a live-observation report for the AIONET Testnet focusing on BEMBH-8192, an 8192-bit memory-bound hash format used within AIONET's validation loop. The primary evidence artifact is a continuous on-screen recording of multi-node operation generating and logging 8192-bit batch hashes while maintaining stable runtime behavior under sustained execution. We clarify what BEMBH-8192 represents, what it does not represent (e.g., arbitrary string generation), and why live observation is a necessary methodology for validating continuity, stability, and honest execution beyond static benchmarks. This report also provides a minimal legacy reference to compute-bound hashing (e.g., SHA families) strictly for scope comparison.

1 Scope and Positioning

This document is a *live observation report* for an AIONET Testnet run using BEMBH-8192. It is not a full protocol specification, tokenomics paper, or formal security proof. The objectives are:

- Establish **what BEMBH-8192 is** in AIONET and the constraints that give it meaning.
- Provide **evidence methodology** centered on continuous live observation (recorded execution).
- Report **observed runtime stability** and operational characteristics of the testnet run.
- Clarify **legacy hash context** (e.g., SHA-256/512) without positioning BEMBH-8192 as a replacement for those primitives.

2 Definition: BEMBH-8192

2.1 Terminology

BEMBH-8192 stands for *8192-Bit Memory-Bound Hash*. In the AIONET Testnet context, BEMBH-8192 is a fixed-width 8192-bit hash output used as a high-resolution batch commitment over structured runtime state and transaction-related inputs produced during live execution.

2.2 Operational Meaning

The meaning of an 8192-bit hash output is not derived from its length alone. Its meaning arises from:

- **Deterministic construction:** a defined input schema and procedure.
- **Execution context:** generation occurs inside a live validator workflow rather than offline string creation.
- **Auditability:** the output is emitted in a log format (e.g., JSONL) with associated metadata (e.g., height, timing).
- **Continuity:** repeated generation over time enables observation of stability and behavior under sustained operation.

3 What BEMBH-8192 Is Not

Because wide hashes can be misunderstood as "just big numbers," we explicitly state what is *not* being claimed:

- Not a claim that "bigger bits automatically means better security" without a threat model.
- Not a random 8192-bit string generator with no structured input meaning.
- Not a benchmark contest for maximum hash throughput.
- Not a GPU-mining primitive or proof-of-work replacement.

4 Legacy Hash Context (SHA Families)

SHA-256 and SHA-512 are widely used cryptographic hash functions designed for strong preimage and collision resistance under a compute-bound model. This report references SHA families only to highlight scope differences:

- **Compute-bound emphasis (legacy):** optimization typically focuses on throughput and resistance properties.
- **Live validation emphasis (AIONET context):** focuses on continuity, stability, and the ability to anchor high-resolution state commitments during ongoing execution.

BEMBH-8192 is presented as a *memory-bound, execution-contextual* hashing format used inside AIONET testnet validation, rather than a general-purpose replacement for SHA primitives.

5 Why Live Observation (Recorded Execution) Matters

Static benchmarks, screenshots, or offline logs can be insufficient to establish certain properties. Live observation is used to support:

- **Continuity:** demonstrating sustained operation across time.
- **Stability:** observing that the system does not degrade or require manual recovery at termination.
- **Honest execution:** reducing the possibility of replayed logs or synthetic "demo" outputs.
- **Operational realism:** capturing real-time generation, logging, and node interactions.

5.1 Recording Compression Note

For distribution and review, the live recording may be time-compressed (constant speed fast-forward) to reduce duration while preserving sequence integrity. Time compression does not change the underlying runtime event order.

6 Testnet Configuration Summary

6.1 Nodes and Roles

This testnet run consists of a coordinator and multiple nodes (including normal and adversarial/spoofing roles). The purpose is to validate multi-node operation under a realistic topology.

Component	Role	Notes
Coordinator	Aggregation / orchestration	Accepts node messages, logs outputs
Node(s)	Normal validator simulation	Generates transactions / state updates
Node(s)	Spoof/adversarial simulation	Used to test detection/robustness paths

Table 1: High-level testnet role summary (fill exact counts/IDs as needed).

6.2 Hash Output Artifact

Each block-height (or step) emits a record containing an 8192-bit hash in hexadecimal representation. Example fields may include:

- `block_height`
- `finality_ms`
- `batch_8192_hex`

Note: This report intentionally avoids embedding large raw hash blobs in the main text. Full artifacts should be provided as attached logs or external references.

7 Observation Log and Evidence Artifacts

7.1 Primary Evidence

The primary evidence artifact is a continuous on-screen recording of the testnet run showing:

- active generation of BEMBH-8192 outputs,
- live runtime metrics (where visible),
- ongoing log growth (e.g., JSONL),
- termination without manual recovery, if applicable.

7.2 Secondary Evidence

Secondary evidence artifacts may include:

- JSONL output logs,
- screenshots of sample hash lines,
- host metrics snapshots (CPU/RAM/GPU).

8 Results (To Be Filled with Run-Specific Numbers)

This section is a structured placeholder for inserting your actual measurements from the run.

8.1 Runtime Stability

- Total runtime observed: _____
- Manual recovery needed at termination: _____ (Yes/No)
- Coordinator responsiveness maintained: _____ (Yes/No)

8.2 Performance Indicators

- Peak CPU utilization observed (approx.): _____
- Typical CPU utilization band (approx.): _____
- RAM utilization observed (approx.): _____
- GPU utilization observed (approx.): _____

8.3 Finality Timing

- Median `finality_ms`: _____
- Typical range: _____

9 Interpretation: Why 8192 Bits Here

Within this testnet scope, the use of 8192-bit batch hashes supports:

- **High-resolution commitments:** larger fixed-width outputs provide a wide commitment space for structured batch state.
- **Future transcription layers:** enabling an interface model where raw data inputs can be deterministically mapped to interpretable outputs ("Raw In" $\rightarrow$ "Transcribed Out").
- **Lower collision concern in practice:** wide commitments reduce accidental collision likelihood in the batch commitment layer, assuming a sound construction.

10 Planned Next Step: Raw-to-Meaning Transcription UI

A near-term objective is to demonstrate an interface pattern:

- Left panel: raw batch inputs or raw hash-anchored records (e.g., 20 entries),
- Right panel: deterministic transcription outputs (e.g., 20 interpreted summaries),

enabling reviewers to validate that BEMBH-8192 outputs are the result of structured runtime state rather than arbitrary strings.

11 Limitations

This report does not claim:

- a complete cryptographic security proof of the construction,
- a formal adversarial model covering all classes of attacks,
- performance generalization to all hardware and workloads.

12 Conclusion

This live observation report documents a sustained AIONET Testnet run generating BEMBH-8192 outputs as part of a memory-bound validation workflow. The recording-based methodology is used to establish continuity and honest execution characteristics that static benchmarks cannot capture. Future work will focus on publishing deterministic raw-to-meaning transcription methods and expanding formal definitions and threat modeling around memory-bound hashing within AIONET.

Artifacts: (add links/filenames here)

- Live recording: _____
- JSONL logs: _____
- Screenshots: _____