ENGINEERING STANDARDS LIBRARY / ARTIFICIAL INTELLIGENCE

Inference Serving, Quantization & KV-Cache Standard

The architecture of LLM inference has failed.

PERMANENT DOCUMENT ID

MYTHOS-AI-0014

PUBLICATION

Candidate Standard
Version 1.0.0-candidate

ASSURANCE PROFILE

Standard / Advanced
High Assurance

PERMANENT PUBLICATION IDENTITY

Inference Serving, Quantization & KV-Cache Standard

DOCUMENT ID
MYTHOS-AI-0014

VERSION
1.0.0-candidate

STATUS
Candidate Standard

PUBLISHER
Mythos Systems

PUBLICATION DATE
2026-07-24

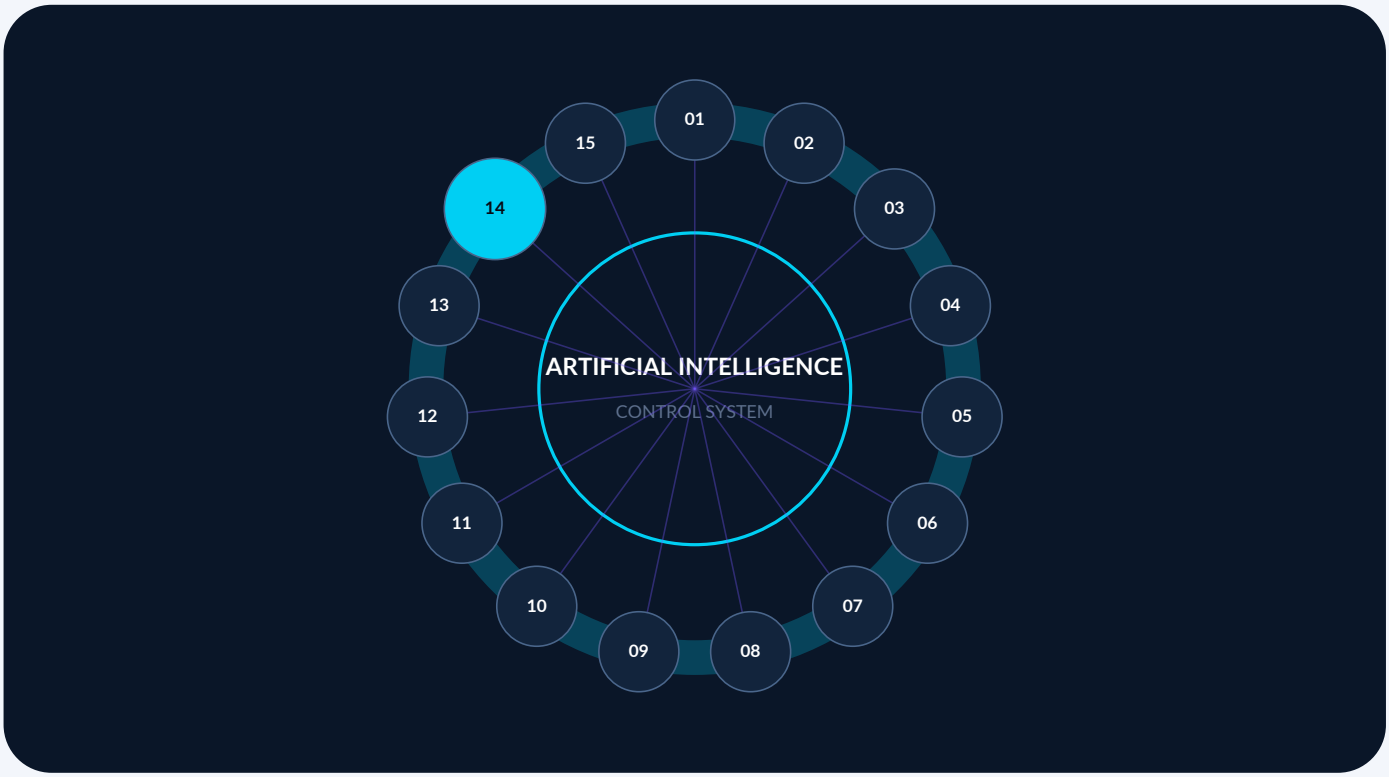
SCHEDULED REVIEW
2027-07-24

11	6	4	1
ATOMIC CRITERIA	ASSURANCE VIEWS	IMPLEMENTATION PROFILES	PERMANENT IDENTITY

Publication doctrine. This standard is a permanent library identity. Interpretation must preserve its recorded evidence, controlled exceptions, formal revision history, and explicit relationship to companion standards.

DOMAIN CONTROL SYSTEM

MYTHOS-AI-0014 inside the artificial intelligence and agentic systems constellation

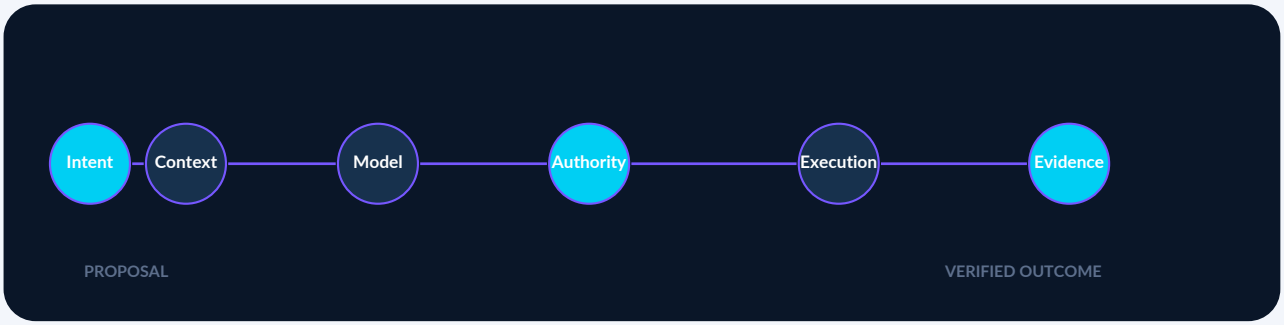


Connected assurance	Architecture, implementation, operations, evidence, and lifecycle are evaluated as one controlled system.
Specialization rule	Companion standards may add precision, but cannot silently weaken this publication.

Executive orientation

Establishes requirements for inference serving, batching, quantization, scheduling, key-value cache management, latency, throughput, isolation, and production reliability.

WHY THIS STANDARD EXISTS	The architecture of LLM inference has failed.
OPERATIONAL SCOPE	This standard covers the evaluation of: - LLM serving engines (vLLM, TensorRT-LLM, llama.cpp, Ollama, LMDeploy) - Continuous batching and dynamic routing algorithms - KV-Cache management (PagedAttention, RadixAttention, CPU offloading) - Quantization paradigms (AWQ, GPTQ, GGUF, FP8, INT4/INT8) - Decoupled Prefill/Decode execution and Speculative Decoding - Thermal and power envelope management for sustained inference
PRIMARY AUDIENCE	- Platform engineers and ML Ops deploying LLM infrastructure - Edge and client engineers building local-first inference (CALLISTO) - Hardware architects evaluating NPU/GPU memory constraints - FinOps teams managing compute and token economics - Standards bodies seeking a reference inference methodology



Assurance principle. Model capability, data quality, identity, authority, operational evidence, and human accountability are treated as a connected system. A high aggregate score cannot compensate for a failed mandatory safety or authority boundary.

Contents

Evaluation criterion index

This standard contains 11 atomic criteria. Each criterion is independently testable and requires retained evidence.

ID	EVALUATION CRITERION
4.1.1	Concurrency Model
4.2.1	VRAM Utilization
4.3.1	Precision vs. Efficiency
4.4.1	Phase Separation
4.5.1	Sustained Performance
4.6.1	Cost Attribution
5.1.1	MISS-Concurrency
5.1.2	MISS-Thermal
6.1.1	VRAM Exhaustion DoS
6.1.2	KV-Cache Side-Channel
6.1.3	Thermal Hijacking

MYTHOS-AI-0014-4.1.1

4.1.1 Concurrency Model

Normative source requirement

How does the server handle multiple concurrent requests?

Score	Criteria
0	Sequential execution (one request at a time)
1	Static batching (waits for batch to fill before executing)
2	Dynamic batching, but stalls on slow requests (head-of-line blocking)
3	Continuous batching: requests join/leave active batches token-by-token
4	Chunked prefill: prefill and decode interleaved in the same batch to maximize GPU Stream Multiprocessor (SM) utilization
5	Perfect throughput: formally verified zero pipeline bubbles, optimal SM occupancy, mathematically proven maximum concurrency

---

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Reproducible benchmark results, workload definitions, raw outputs, and variance records. Model and dataset cards, lineage, licenses, evaluation reports, release approvals, and rollback artifacts.
ASSESSMENT METHOD	MATURITY SIGNAL
Run a version-pinned workload with representative inputs, record distributions rather than a single score, repeat under failure and load, and compare reproduced results with the stated acceptance threshold.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-4.2.1

4.2.1 VRAM Utilization

Normative source requirement

How is the Key-Value cache stored and managed in GPU memory?

Score	Criteria
0	Contiguous memory allocation per request (severe fragmentation, instant OOM)
1	Basic memory pooling, but cannot handle variable sequence lengths efficiently
2	RadixAttention or similar tree-based caching for prefix sharing
3	PagedAttention: KV-cache stored in non-contiguous virtual blocks, zero fragmentation
4	Hierarchical KV offloading: inactive blocks moved to CPU RAM / NVMe SSD transparently
5	Perfect management: formally verified memory bounds, zero OOM risk, distributed KV-cache via RDMA across multiple GPUs

---

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Reproducible benchmark results, workload definitions, raw outputs, and variance records. Context manifests, retrieval traces, source citations, freshness records, conflict decisions, and memory provenance.
ASSESSMENT METHOD	MATURITY SIGNAL
Run a version-pinned workload with representative inputs, record distributions rather than a single score, repeat under failure and load, and compare reproduced results with the stated acceptance threshold.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-4.3.1

4.3.1 Precision vs. Efficiency

Normative source requirement

Is the model quantized to match the hardware memory and compute constraints?

Score	Criteria
0	Unquantized FP32/FP16 models deployed on edge/consumer hardware
1	Basic Post-Training Quantization (PTQ) with high accuracy degradation
2	INT8 quantization used for datacenter GPUs
3	AWQ or GPTQ (INT4) used for edge/local-first hardware with negligible accuracy loss
4	FP8 utilized on next-gen GPUs (Hopper/Blackwell) for hardware-native acceleration
5	Perfect mapping: formally verified accuracy parity, dynamic precision switching, zero thermal violations

---

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Reproducible benchmark results, workload definitions, raw outputs, and variance records. Context manifests, retrieval traces, source citations, freshness records, conflict decisions, and memory provenance.
ASSESSMENT METHOD	MATURITY SIGNAL
Run a version-pinned workload with representative inputs, record distributions rather than a single score, repeat under failure and load, and compare reproduced results with the stated acceptance threshold.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-4.4.1

4.4.1 Phase Separation

Normative source requirement

Are the compute-heavy prefill and memory-heavy decode phases optimized independently?

Score	Criteria
0	Prefill and decode mixed in the same execution batch (causes pipeline stalls)
1	Chunked prefill implemented to prevent decode starvation
2	Basic scheduling prioritizing decode latency over prefill throughput
3	Disaggregated inference: prefill routed to high-compute nodes, decode routed to high-memory nodes
4	Speculative decoding: draft model generates tokens, target model verifies in parallel
5	Perfect optimization: formally verified phase isolation, zero decode latency spikes, mathematical proof of optimal token acceptance rate

---

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Reproducible benchmark results, workload definitions, raw outputs, and variance records. Context manifests, retrieval traces, source citations, freshness records, conflict decisions, and memory provenance.
ASSESSMENT METHOD	MATURITY SIGNAL
Run a version-pinned workload with representative inputs, record distributions rather than a single score, repeat under failure and load, and compare reproduced results with the stated acceptance threshold.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-4.5.1

4.5.1 Sustained Performance

Normative source requirement

Can the inference server sustain maximum load without thermal failure?

Score	Criteria
0	Throttles and crashes under sustained 100% GPU load
1	Thermal throttling reduces clock speeds, causing unpredictable latency
2	Power limits hardcoded to prevent throttling, but wastes peak capacity
3	Dynamic power scaling: server monitors thermal sensors and throttles batch size proactively
4	NPU/GPU heterogeneous offloading: compute split across silicon to balance thermals
5	Perfect resilience: formally verified thermal bounds, zero latency degradation over 24h soak tests, mathematically proven power-to-token efficiency

---

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Reproducible benchmark results, workload definitions, raw outputs, and variance records.
ASSESSMENT METHOD	MATURITY SIGNAL
Run a version-pinned workload with representative inputs, record distributions rather than a single score, repeat under failure and load, and compare reproduced results with the stated acceptance threshold.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-4.6.1

4.6.1 Cost Attribution

Normative source requirement

Is inference cost measured and governed at the token level?

Score	Criteria
0	No visibility into cost-per-token or compute utilization
1	Basic metrics on total tokens generated
2	Per-request token tracking, but no GPU cost attribution
3	Real-time FinOps: cost-per-token calculated based on GPU lease rates and power draw
4	Automated budget enforcement: requests dropped or down-routed if user token budget exceeded
5	Perfect economics: formally verified cost attribution, AI-driven workload routing to cheapest available compute, zero wasted GPU cycles

---

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Reproducible benchmark results, workload definitions, raw outputs, and variance records.
ASSESSMENT METHOD	MATURITY SIGNAL
Run a version-pinned workload with representative inputs, record distributions rather than a single score, repeat under failure and load, and compare reproduced results with the stated acceptance threshold.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-5.1.1

5.1.1 MISS-Concurrency

Normative source requirement

• OOM Injection: 100+ tests sending 1,000 concurrent requests with 128k-token contexts, verifying the server gracefully offloads or queues rather than crashing with OOM.

• Variable Length Stress: 50+ tests mixing 1-token requests with 10,000-token requests in the same batch, verifying continuous batching does not starve short requests.

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Context manifests, retrieval traces, source citations, freshness records, conflict decisions, and memory provenance.
ASSESSMENT METHOD	MATURITY SIGNAL
Inject stale, conflicting, low-authority, and malicious information; inspect selection and citation behavior; verify scope isolation, correction, deletion, and provenance retention.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-5.1.2

5.1.2 MISS-Thermal

Normative source requirement

• 24h Soak Test: 100+ tests running continuous inference at max throughput for 24 hours, verifying the GPU does not throttle and latency percentiles (p99) remain stable.

• KV Eviction Test: 50+ tests forcing the server to evict inactive KV-caches to CPU RAM, verifying the context can be seamlessly recalled when the user resumes interaction.

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Reproducible benchmark results, workload definitions, raw outputs, and variance records. Context manifests, retrieval traces, source citations, freshness records, conflict decisions, and memory provenance.
ASSESSMENT METHOD	MATURITY SIGNAL
Run a version-pinned workload with representative inputs, record distributions rather than a single score, repeat under failure and load, and compare reproduced results with the stated acceptance threshold.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-6.1.1

6.1.1 VRAM Exhaustion DoS

Normative source requirement

Severity: Critical Description: An attacker (or a runaway agent) submits 1,000 concurrent requests with massive context windows, intentionally exhausting GPU VRAM and crashing the inference server for all users.

Required Controls: 1. PagedAttention to minimize memory waste. 2. Strict per-user/request concurrency limits and max-sequence-length bounds. 3. Graceful KV-cache eviction policies (LRU) rather than hard OOM crashes.

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Context manifests, retrieval traces, source citations, freshness records, conflict decisions, and memory provenance.
ASSESSMENT METHOD	MATURITY SIGNAL
Inject stale, conflicting, low-authority, and malicious information; inspect selection and citation behavior; verify scope isolation, correction, deletion, and provenance retention.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-6.1.2

6.1.2 KV-Cache Side-Channel

Normative source requirement

Severity: High Description: An attacker probes the inference server to infer the contents of another user's prompt by measuring the latency of requests that share a common prefix in the KV-cache (RadixAttention cache hits are faster).

Required Controls: 1. Strict isolation of KV-caches per tenant/user. 2. Avoid sharing prefix caches across different security contexts.

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Reproducible benchmark results, workload definitions, raw outputs, and variance records. Policy configuration, identity or trust records, authorization decisions, revocation evidence, and adversarial test results.
ASSESSMENT METHOD	MATURITY SIGNAL
Run a version-pinned workload with representative inputs, record distributions rather than a single score, repeat under failure and load, and compare reproduced results with the stated acceptance threshold.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

MYTHOS-AI-0014-6.1.3

6.1.3 Thermal Hijacking

Normative source requirement

Severity: Medium Description: A malicious workload intentionally maximizes GPU compute in a loop, attempting to physically damage the hardware or trigger a system-wide thermal shutdown.

Required Controls: 1. Hardware-enforced power limits (TDP). 2. Inference server monitoring of board temperatures with automated circuit-breaking.

CONTROL INTENT	REQUIRED EVIDENCE
Create a measurable, reviewable control that can be verified independently of model or vendor claims.	Architecture records, implemented configuration, test evidence, operational telemetry, exception decisions, and accountable approval.
ASSESSMENT METHOD	MATURITY SIGNAL
Inspect the architecture and configured control, execute normal and adverse scenarios, verify measurable outcomes, sample retained evidence, and confirm that exceptions are time-bound and approved.	0 absent | 1 ad hoc | 2 repeatable | 3 controlled | 4 measured | 5 continuously verified

Decision rule. A criterion is not satisfied by documentation alone when the claim concerns runtime behavior. Reproduced evidence, responsible ownership, known limitations, and controlled exceptions are required.

Current authority and freshness register

These sources inform interpretation but do not convert this candidate publication into certification, legal compliance, or implementation evidence. Rolling sources must be version-pinned at adoption time.

AUTHORITY	VERSION / FRESHNESS	APPLICABILITY LIMIT
NIST - Artificial Intelligence Risk Management Framework (AI RMF 1.0)	NIST AI 100-1; current framework, revision underway Expires 2026-10-24	Voluntary risk-management framework; does not establish product certification or implementation effectiveness.
NIST - Generative Artificial Intelligence Profile	NIST AI 600-1, July 2024 Expires 2027-01-24	Profile guidance requires tailoring to the organization, use case, and risk tolerance.
NIST - Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations	NIST AI 100-2e2025 Expires 2027-01-24	Taxonomy and terminology do not substitute for system-specific red-team evidence.
OWASP - Top 10 for LLM Applications 2025	2025 edition Expires 2026-10-24	Community risk guidance; prioritization and mitigations require application-specific validation.
OWASP - Top 10 for Agentic Applications 2026	2026 edition Expires 2026-10-24	Emerging community guidance for agentic systems; not a certification framework.
OpenTelemetry - Semantic Conventions	1.43.0 rolling specification; GenAI conventions maintained separately Expires 2026-08-24	Several GenAI and agent conventions remain under active development and require version pinning.
C2PA - Content Credentials Technical Specification	2.4 rolling publication Expires 2026-10-24	Provenance establishes signed assertions and tamper evidence, not the truth or quality of the underlying content.

Source lineage appendix

The following text preserves the substantive source draft in a normalized public form. Where it conflicts with the permanent publication identity, candidate status, current authority register, or evidence requirements above, the controlled publication layer governs.

0. Executive Mandate

The architecture of LLM inference has failed.

Organizations attempt to serve production Large Language Models by wrapping Hugging Face `transformers` pipelines in a FastAPI server. They process one request at a time, loading the model weights and computing the attention matrix sequentially. When concurrent requests hit the server, the GPU VRAM fragments, the KV-cache overflows into system RAM (causing catastrophic latency spikes), and the system grinds to a halt.

This standard exists because:

1. Sequential Inference is Economically Unviable: An LLM serving pipeline that cannot process multiple concurrent requests in a single batch wastes 90% of the GPU's parallel compute capabilities. Production serving MUST utilize continuous (in-flight) batching and PagedAttention.

2. Unmanaged KV-Cache is an OOM Trap: As context windows grow to 128k+ tokens, the Key-Value cache expands exponentially. Naive memory allocation causes fragmentation and Out-Of-Memory (OOM) errors. Systems MUST manage the KV-cache using OS-style virtual memory paging (e.g., vLLM's PagedAttention) and offload to CPU/NVMe when capacity is reached.

3. Unquantized Models are Thermally Irresponsible: Deploying unquantized FP16 models on edge devices or consumer GPUs guarantees thermal throttling and power exhaustion. Systems MUST implement hardware-aware quantization (INT4/INT8 via AWQ, GPTQ, or GGUF) with negligible accuracy degradation.

4. Prefill and Decode Must Be Decoupled: The prefill phase (processing the prompt) is compute-bound, while the decode phase (generating tokens) is memory-bound. Mixing them in the same batch causes pipeline stalls. Systems MUST utilize disaggregated inference or chunked prefill to optimize both phases independently.

This document establishes the Mythos Inference Serving Standard (MISS), a comprehensive, evidence-based methodology for evaluating LLM serving architectures against memory efficiency, throughput optimization, and thermal resilience.

Part I. Scope and Applicability

1.1 In Scope

This standard covers the evaluation of:

- LLM serving engines (vLLM, TensorRT-LLM, llama.cpp, Ollama, LMDeploy)
- Continuous batching and dynamic routing algorithms
- KV-Cache management (PagedAttention, RadixAttention, CPU offloading)
- Quantization paradigms (AWQ, GPTQ, GGUF, FP8, INT4/INT8)
- Decoupled Prefill/Decode execution and Speculative Decoding
- Thermal and power envelope management for sustained inference

1.2 Out of Scope

- General AI agent orchestration (covered in MYTHOS-AI-0001)
- General sovereign/air-gapped deployment (covered in MYTHOS-EMT-0002)
- General GPU cluster topology (covered in MYTHOS-HW-0001)

1.3 Audience

- Platform engineers and ML Ops deploying LLM infrastructure
- Edge and client engineers building local-first inference (CALLISTO)
- Hardware architects evaluating NPU/GPU memory constraints
- FinOps teams managing compute and token economics
- Standards bodies seeking a reference inference methodology

Part II. Definitions and Taxonomy

2.1 Inference Dimensions

Dimension	Definition
Continuous Batching	An execution paradigm where new requests are injected into an active batch on every token generation step, eliminating pipeline bubbles and maximizing GPU utilization.
PagedAttention	An attention algorithm inspired by OS virtual memory, storing the KV-cache in non-contiguous blocks to eliminate memory fragmentation and waste.
KV-Cache Offloading	The practice of moving inactive Key-Value caches from GPU VRAM to CPU RAM or NVMe SSDs to free up high-speed memory for active contexts.
Disaggregated Inference	An architecture that physically separates the prefill (prompt processing) compute cluster from the decode (token generation) cluster.
Speculative Decoding	A technique using a small, fast "draft" model to generate candidate tokens, which are then verified in parallel by the larger "target" model, reducing latency.

2.2 The Inference Serving Doctrine

> A sequential inference pipeline is a waste of silicon. Unquantized FP16 on the edge is a space heater. A KV-cache without paging is a memory leak. If the server cannot survive 1,000 concurrent requests without OOM, it is a prototype.

Part III. Evaluation Methodology

3.1 Scoring Model

Each capability is scored from 0 to 5.

Score	Meaning
0	Fails basic task; sequential execution, FP16, OOMs under load
1	Basic batching, but severe VRAM fragmentation and no quantization
2	Functional serving, but lacks PagedAttention or decode optimization
3	Solid production performance; continuous batching, PagedAttention, AWQ/INT8
4	Strong performance; KV offloading, disaggregated prefill, speculative decoding
5	Best-in-class; sets the standard for mathematically verified, zero-fragmentation inference

Weighting

Category	Weight	Rationale
Continuous Batching & Throughput	20%	Sequential processing makes LLMs economically unviable
KV-Cache Memory Management	20%	Unmanaged VRAM guarantees OOM crashes on long contexts
Quantization & Hardware Mapping	20%	FP16 is thermally and economically unsustainable at scale
Prefill/Decode Optimization	15%	Mixing compute-bound and memory-bound phases causes stalls
Thermal & Power Resilience	15%	Sustained inference must respect hardware envelope limits
Token Economics & FinOps	10%	Inference must be measurable and budgetable

Total: 100%

A system MUST score at least 3.0 in Continuous Batching and KV-Cache Management to be considered for production use.

Part IV. Evaluation Categories

4.1 Continuous Batching and Throughput (Weight: 20%)

4.1.1 Concurrency Model

How does the server handle multiple concurrent requests?

Score	Criteria
0	Sequential execution (one request at a time)
1	Static batching (waits for batch to fill before executing)
2	Dynamic batching, but stalls on slow requests (head-of-line blocking)
3	Continuous batching: requests join/leave active batches token-by-token
4	Chunked prefill: prefill and decode interleaved in the same batch to maximize GPU Stream Multiprocessor (SM) utilization
5	Perfect throughput: formally verified zero pipeline bubbles, optimal SM occupancy, mathematically proven maximum concurrency

4.2 KV-Cache Memory Management (Weight: 20%)

4.2.1 VRAM Utilization

How is the Key-Value cache stored and managed in GPU memory?

Score	Criteria
0	Contiguous memory allocation per request (severe fragmentation, instant OOM)
1	Basic memory pooling, but cannot handle variable sequence lengths efficiently
2	RadixAttention or similar tree-based caching for prefix sharing
3	PagedAttention: KV-cache stored in non-contiguous virtual blocks, zero fragmentation
4	Hierarchical KV offloading: inactive blocks moved to CPU RAM / NVMe SSD transparently
5	Perfect management: formally verified memory bounds, zero OOM risk, distributed KV-cache via RDMA across multiple GPUs

4.3 Quantization and Hardware Mapping (Weight: 20%)

4.3.1 Precision vs. Efficiency

Is the model quantized to match the hardware memory and compute constraints?

Score	Criteria
0	Unquantized FP32/FP16 models deployed on edge/consumer hardware
1	Basic Post-Training Quantization (PTQ) with high accuracy degradation
2	INT8 quantization used for datacenter GPUs
3	AWQ or GPTQ (INT4) used for edge/local-first hardware with negligible accuracy loss

Score	Criteria
4	FP8 utilized on next-gen GPUs (Hopper/Blackwell) for hardware-native acceleration
5	Perfect mapping: formally verified accuracy parity, dynamic precision switching, zero thermal violations

4.4 Prefill/Decode Optimization (Weight: 15%)

4.4.1 Phase Separation

Are the compute-heavy prefill and memory-heavy decode phases optimized independently?

Score	Criteria
0	Prefill and decode mixed in the same execution batch (causes pipeline stalls)
1	Chunked prefill implemented to prevent decode starvation
2	Basic scheduling prioritizing decode latency over prefill throughput
3	Disaggregated inference: prefill routed to high-compute nodes, decode routed to high-memory nodes
4	Speculative decoding: draft model generates tokens, target model verifies in parallel
5	Perfect optimization: formally verified phase isolation, zero decode latency spikes, mathematical proof of optimal token acceptance rate

4.5 Thermal and Power Resilience (Weight: 15%)

4.5.1 Sustained Performance

Can the inference server sustain maximum load without thermal failure?

Score	Criteria
0	Throttles and crashes under sustained 100% GPU load
1	Thermal throttling reduces clock speeds, causing unpredictable latency
2	Power limits hardcoded to prevent throttling, but wastes peak capacity
3	Dynamic power scaling: server monitors thermal sensors and throttles batch size proactively
4	NPU/GPU heterogeneous offloading: compute split across silicon to balance thermals
5	Perfect resilience: formally verified thermal bounds, zero latency degradation over 24h soak tests, mathematically proven power-to-token efficiency

4.6 Token Economics and FinOps (Weight: 10%)

4.6.1 Cost Attribution

Is inference cost measured and governed at the token level?

Score	Criteria
0	No visibility into cost-per-token or compute utilization
1	Basic metrics on total tokens generated
2	Per-request token tracking, but no GPU cost attribution
3	Real-time FinOps: cost-per-token calculated based on GPU lease rates and power draw
4	Automated budget enforcement: requests dropped or down-routed if user token budget exceeded

Score	Criteria
5	Perfect economics: formally verified cost attribution, AI-driven workload routing to cheapest available compute, zero wasted GPU cycles

Part V. The Mythos Inference Serving Suite (MISS)

Traditional AI benchmarks measure zero-shot accuracy. The MISS evaluates memory fragmentation, concurrency survival, and thermal resilience.

5.1 Suite Composition

5.1.1 MISS-Concurrency

• OOM Injection: 100+ tests sending 1,000 concurrent requests with 128k-token contexts, verifying the server gracefully offloads or queues rather than crashing with OOM.

• Variable Length Stress: 50+ tests mixing 1-token requests with 10,000-token requests in the same batch, verifying continuous batching does not starve short requests.

5.1.2 MISS-Thermal

• 24h Soak Test: 100+ tests running continuous inference at max throughput for 24 hours, verifying the GPU does not throttle and latency percentiles (p99) remain stable.

• KV Eviction Test: 50+ tests forcing the server to evict inactive KV-caches to CPU RAM, verifying the context can be seamlessly recalled when the user resumes interaction.

5.2 Execution Protocol

```
1. Deploy inference server (vLLM/TensorRT-LLM) with target hardware (GPU/NPU)
2. Run MISS suite (automated concurrency stress + thermal profiling)
3. Score each category 0-5
4. Check for sequential execution or unmanaged VRAM allocation (instant disqualification)
5. If all thresholds met: approve for production
```

Part VI. Security Threat Model for Inference Serving

6.1 Threat Catalog

6.1.1 VRAM Exhaustion DoS

Severity: Critical Description: An attacker (or a runaway agent) submits 1,000 concurrent requests with massive context windows, intentionally exhausting GPU VRAM and crashing the inference server for all users.

Required Controls:

1. PagedAttention to minimize memory waste.

2. Strict per-user/request concurrency limits and max-sequence-length bounds.

3. Graceful KV-cache eviction policies (LRU) rather than hard OOM crashes.

6.1.2 KV-Cache Side-Channel

Severity: High Description: An attacker probes the inference server to infer the contents of another user's prompt by measuring the latency of requests that share a common prefix in the KV-cache (RadixAttention cache hits are faster).

Required Controls:

1. Strict isolation of KV-caches per tenant/user.

2. Avoid sharing prefix caches across different security contexts.

6.1.3 Thermal Hijacking

Severity: Medium Description: A malicious workload intentionally maximizes GPU compute in a loop, attempting to physically damage the hardware or trigger a system-wide thermal shutdown.

Required Controls:

1. Hardware-enforced power limits (TDP).

2. Inference server monitoring of board temperatures with automated circuit-breaking.

Part VII. The Mythos Inference Serving Standard

7.1 The Mythos Inference Doctrine

```
A sequential inference pipeline is a waste of silicon.
Unquantized FP16 on the edge is a space heater.

A KV-cache without paging is a memory leak.
If the server cannot survive 1,000 concurrent requests without OOM, it is a prototype.

Compute may be parallel.
Memory management must be absolute.
```

7.2 Selection Rules

1. No inference architecture enters production without passing MISS.
2. No architecture is approved if it processes requests sequentially.
3. No architecture is approved without PagedAttention or equivalent KV-cache management.
4. No edge architecture is approved if it relies on unquantized (FP16/FP32) models.
5. No architecture is approved if it mixes prefill and decode in the same execution batch.

7.3 The Prime Directive for Inference Architecture

> Batch the token, do not process the sequence. > Page the cache, do not fragment the VRAM. > Quantize the weight, do not throttle the GPU. > Disaggregate the phase, do not stall the pipeline.

```
Academic benchmarks measure zero-shot accuracy.
Applied benchmarks measure continuous batching throughput.
Security benchmarks measure VRAM isolation.
Operational benchmarks measure thermal resilience.

> Context may be infinite.
> VRAM must be absolute.
```

End of Standard MYTHOS-AI-0014

End of controlled publication

MYTHOS-AI-0014 remains a Candidate Standard until technical review, security and privacy review, accessibility review, current-source validation, and formal approval are recorded.