

Juniper® Validated Design JVD Test Report Brief: AI Data Center Frontend Fabric for Inference with HPE Juniper QFX switches, Apstra Data Center Director and AMD Instinct MI300X GPUs



JVD-AICLUSTERDC-AIMLINF-01-01

Introduction

This document summarizes the validation testing performed for the AI Data Center Frontend Fabric for Inference solution using HPE Juniper Networks QFX5k switches, HPE Juniper Apstra Data Center Director, and AMD Instinct MI300X GPU systems.

The purpose of the testing is to characterize high-performance AI inference behavior across a standards-based frontend IP Ethernet fabric. The validation covers both Single-node and Multi-node inference scenarios. The Single-node scenario uses SGLang model-serving endpoints running on AMD MI300X systems, while the Multi-node scenario uses SGLang model-serving endpoints with request distribution through Envoy. In both scenarios, inference request and response traffic is generated by NVIDIA GenAI-Perf, and frontend fabric behavior is observed during benchmark execution.

Test Goals

The goals of this validation are:

1. Characterize inference performance across the validated frontend IP fabric.
2. Validate the QFX5140-24CD80 as a frontend leaf platform for production AI inference deployments.
3. Measure inference-serving metrics, including Time to First Token, Inter-Token Latency, Tokens per Second, End-to-End Request Latency, concurrency, and request success rate.

Non-Goals:

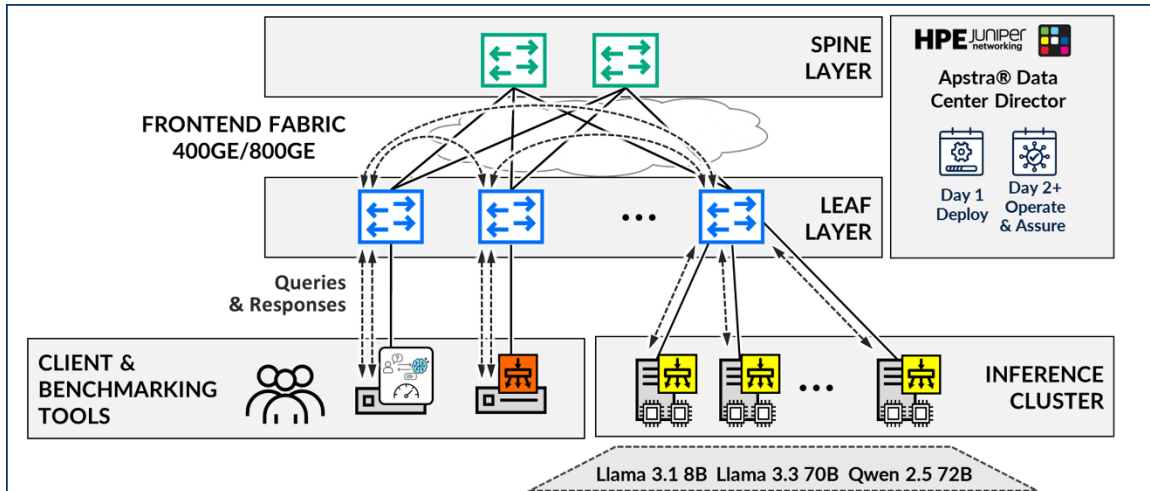
The following items are outside the scope of this test report:

1. GPU Backend fabric validation.
2. Storage Backend fabric validation.
3. DCQCN, PFC, ECN, and fabric load balancing tuning for frontend inference traffic.

Test Environment Components

The test environment used for testing is depicted in Figure 1.

Figure 1. Test Environment Overview



The test environment includes the following components:

Table 1: Frontend Fabric Components

Validated Component	Role in Solution	Description
HPE Juniper Networks QFX switches	Frontend leaf nodes	Provide frontend connectivity for inference servers and client systems.
HPE Juniper Networks QFX switches	Frontend spine nodes	Provide redundant high-speed connectivity across the three-stage Clos frontend fabric.
HPE Juniper Apstra Data Center Director	Fabric automation	Automates fabric deployment and supports operational consistency.
AMD Instinct MI300X GPU servers x 2	Inference compute nodes	Host SGLang model-serving endpoints on GPU-backed worker processes.
ConnectX-7 NICs	MI300X frontend NICs	Provide 400G frontend connectivity from the MI300X servers to the QFX fabric.
Lambda Scaler systems x 2	Client and service hosts	Provide hosts for GenAI-Perf benchmark generation and Envoy load balancing.
- Lambda-Scaler-01	GenAI-Perf host	Runs NVIDIA GenAI-Perf in a container to generate inference traffic and collect metrics.

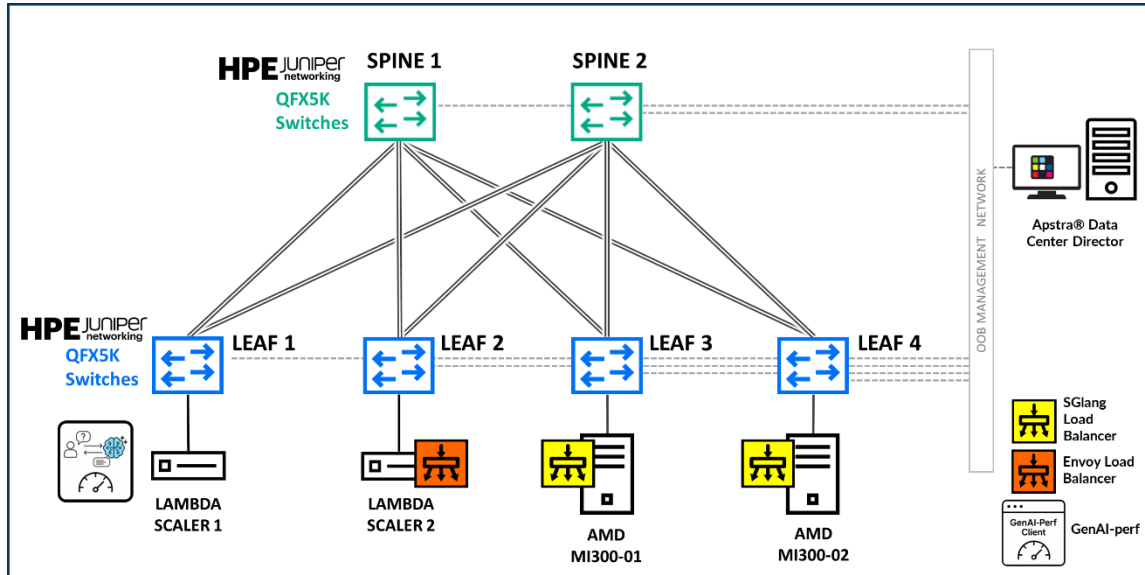
- Lambda-Scaler-02	Envoy host	Runs Envoy in a container to distribute requests across MI300X inference servers.
Envoy	Load balancer	Provides scale-out request distribution across SGLang inference endpoints.
SGLang	Inference serving framework	Loads validated LLMs and runs GPU-backed model-serving workers on the MI300X systems.
SGLang Router	Request routing	Receives inference requests and routes them to local SGLang worker processes.
NVIDIA GenAI-Perf	Benchmark tool	Generates inference load and collects benchmark performance metrics.

Refer to [Appendix A – Table A1](#) for the validated software versions and additional environment details.

Validated Test Topology

The validated topology includes four leaf nodes and two spine nodes in a three-stage Clos frontend IP fabric, as shown in Figure 2.

Figure 2. AI Inference Frontend Fabric Topology



Validated QFX Platforms

The following platforms were validated to act as leaf and spine nodes in the Inference Frontend Fabric described in the previous section.

Table 2: Validated QFX Platforms

VALIDATED LEAF NODES	VALIDATED SPINE NODES
HPE Juniper QFX5130-32CD	HPE Juniper QFX5220-32CD
HPE Juniper QFX5140-24CD8O	HPE Juniper QFX5230-64CD
HPE Juniper QFX5240-64OD	HPE Juniper QFX5240-64OD

Each frontend leaf connects to both spine nodes using 2 x 400GbE Ethernet links, providing redundant and scalable connectivity across the frontend fabric. The AMD Instinct MI300X GPU servers connect to the leaf using 400GbE Ethernet links with ConnectX-7 NICs. The Lambda Scaler systems running Envoy and GenAI-Perf connect to leaf nodes using 100GbE Ethernet links with ConnectX-6 NICs.

Note: Additional NICs and link speeds may be added in future updates of this JVD.

HPE Juniper Apstra Data Center Director creates and deploys the configuration to each device in the fabric.

Refer to [Appendix B](#) for topology implementation details, Apstra fabric automation, and eBGP behavior.

Validated Test Scenarios

The benchmark plan evaluates two primary inference scenarios, as shown in Figure 3.

Figure 3. Test Scenarios – Single-node vs. Multi-node

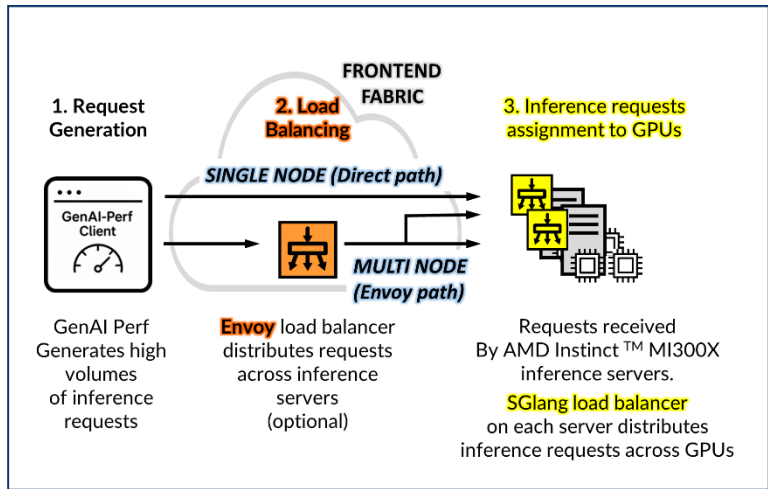


Table 3: Testing Scenarios

Test Scenario	Traffic Path
---------------	--------------

Single-node (direct path)	GenAI-Perf sends requests directly to the SGLang Router running on one of the AMD MI300X servers in the test topology. The SGLang Router on that server distributes the load across the GPUs.
Multi-node (Envoy load balanced path)	GenAI-Perf sends requests to an Envoy frontend endpoint. Envoy distributes the requests across the two AMD MI300X servers running SGLang in the test topology. The SGLang Router on each server distributes the load across GPUs.

Refer to [Appendix C](#) for more details on [Multi-node SGLang](#), [Multi-node Envoy Load balanced for Inference](#), and [SGLang Router](#).

Validated AI Models

The following AI models were evaluated.

Table 4: AI Models Tested

Model	Precision	Context	Purpose
Llama 3.1 8B	FP16	Short	Smaller-model baseline for responsiveness, throughput, and concurrency behavior.
Llama 3.1 8B	FP16	Long	Long-context RAG-style summary workload.
Llama 3.3 70B	FP16	Short	Large-model baseline for compute, memory, and latency characterization.
Llama 3.3 70B	FP16	Long	Large-model long-context workload used to evaluate prefill latency and sustained generation behavior.
Llama 3.3 70B	FP8	Short	Llama FP8 performance profile used to characterize optimized throughput behavior.
Llama 3.3 70B	FP8	Long	Llama FP8 long-context workload used to evaluate optimized performance with larger prompts.
Qwen 2.5 72B	FP8	Short	Qwen FP8 performance profile for throughput characterization.
Qwen 2.5 72B	FP8	Long	Long-context RAG-style summary workload.
Qwen 2.5 72B	FP16	Short	Large-model baseline.
Qwen 2.5 72B	FP16	Long	Optional long-context baseline.

Using different model precision (FP8 vs FP16) and context profiles (short vs long) allows inference behavior characterization under different workload conditions.

Refer to [Appendix C Table C3](#) and [Table C4](#) to learn more about the model precision, and context profiles included.

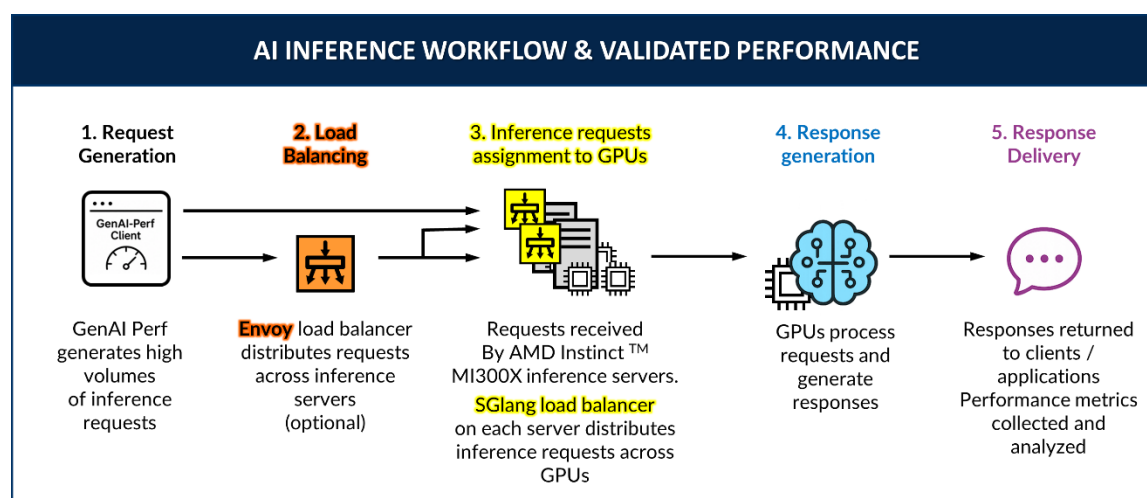
Test Methodology Overview

Benchmark inference requests were generated from the client host using GenAI-Perf. Requests were sent either directly to one AMD MI300X server or through the Envoy load balancer, as shown in Figure 4. After the requests were processed by the servers and GPUs, the responses and benchmark metrics were collected and analyzed.

Each validated model listed in [Table 4](#) was tested over both the direct Single-node path and the Multi-node Envoy path.

Examples of the GenAI-Perf commands used to generate requests are provided in [Appendix D - Table D1](#).

Figure 4. Benchmark Inference Workflow



The values for [concurrency](#), [request count](#), [input sequence length \(ISL\)](#), [output sequence length \(OSL\)](#), and [warm-up](#) requests were adjusted to optimize each test scenario and model. The primary objective was to minimize **Time to First Token (TTFT)**.

Definitions for these parameters can be found in [Appendix D - Table D3](#) and the specific values used for each test case can be found in [Appendix D - Table D4](#).

For each test case, the following metrics were collected and analyzed: [Time to First Token \(TTFT\)](#), [Request Latency](#), [Inter-Token Latency \(ITL\)](#), [Output Tokens per Second \(TPS\)](#), and [request throughput](#). Metric definitions are provided in [Appendix D - Table D5](#).

Inference Performance Results

The test results for each scenario are summarized in Table 5.

Table 5: Inference Performance Results Summary

Test Case ID	Path	Model	TTFT [ms]	Request Latency [ms]	ITL [ms]	Output TPS [tokens/sec]	Request throughput [req/s]
T1	Single-node	Llama 3.1 8B FP16, short	190.08	3173.36	27.43	85721.93	776.89
T2	Multi-node	Llama 3.1 8B FP16, short	442.04	3093.96	24.37	177829.57	1609.84
T3	Single-node	Llama 3.1 8B FP16, long	286.22	5083.3	20.95	51727.87	224.33
T4	Multi-node	Llama 3.1 8B FP16, long	347.73	5469.77	22.47	106404.06	463.66
T5	Single-node	Llama 3.3 70B FP8, short	302.97	7834.2	66.39	19262.77	167.85
T6	Multi-node	Llama 3.3 70B FP8, short	309.99	7508.15	63.41	33646.61	292.72
T7	Single-node	Llama 3.3 70B FP8, long	307.94	10805.95	45.25	8225	35.26
T8	Multi-node	Llama 3.3 70B FP8, long	299.63	11571.46	48.51	17350.18	74.27

T9	Single-node	Llama 3.3 70B FP16, short	303.24	10527.16	90.54	20851.89	182.45
T10	Multi-node	Llama 3.3 70B FP16, short	296.62	8921.99	76.26	33846.52	295.58
T11	Single-node	Llama 3.3 70B FP16, long	1581.2	11854.66	44.25	2389.38	10.25
T12	Multi-node	Llama 3.3 70B FP16, long	1627.09	11303.53	41.67	4163.57	17.84
T13	Single-node	Qwen 2.5 72B FP16, short	355	10998.06	93.56	17400.53	151.32
T14	Multi-node	Qwen 2.5 72B FP16, short	351.24	8342.92	70.27	33016.8	287.21
T15	Single-node	Qwen 2.5 72B FP16, long	2767.29	13478.27	46.59	1701.57	7.34
T16	Multi-node	Qwen 2.5 72B	1694.95	11845.93	44.14	4490.07	19.17

		FP16, long					
T17	Single-node	Qwen 2.5 72B FP8, short	449.9	9876.75	90.51	20302.38	188.83
T18	Multi-node	Qwen 2.5 72B FP8, short	296.85	7619.54	70.48	32073.03	297.49
T19	Single-node	Qwen 2.5 72B FP8, long	1821.54	9182.83	55.77	1862.82	10.27
T20	Multi-node	Qwen 2.5 72B FP8, long	887.67	7122.58	38.52	4969.92	27.11

Note: In Table 5, Multi-node refers to the direct SGLang test path using one AMD Instinct MI300X server from the validated lab topology. Multi-node refers to the Envoy load balanced test path using both AMD Instinct MI300X servers in the validated lab topology.

Results Analysis

The benchmark results in Table 5 characterize inference-serving behavior across the Llama 3.1 8B, Llama 3.3 70B, and Qwen 2.5 72B model families using AMD Instinct MI300X systems running SGLang. The results compare Single-node SGLang inference against Multi-node Envoy load balanced inference, with NVIDIA GenAI-Perf used to generate inference requests and collect performance metrics. The test set includes short-context and long-context profiles, FP16 and FP8 precision formats, and multiple model sizes.

The results also show that the validated frontend network supported inference interactions between the GenAI-Perf client, Envoy load balancer, and SGLang inference servers across both Single-node and Multi-node scenarios. The frontend fabric provided the IP connectivity required for request generation, request distribution, response delivery, and performance metric collection. The benchmark results did not show frontend-fabric behavior that prevented successful benchmark execution or scale-out comparison.

The analysis focuses primarily on two metrics: Time to First Token (TTFT) and output Tokens per Second (TPS). TTFT is the most visible responsiveness metric because it represents how long the user waits before the model begins generating a response. For interactive inference services, the first generated token strongly influences whether the system feels responsive. TPS is the primary serving-capacity metric because it measures how efficiently the inference environment generates output tokens under load. Together, TTFT and TPS provide the most useful view of this JVD's

objectives: validating a frontend Ethernet fabric that can support responsive, high-throughput AI inference traffic across Single-node and scale-out Multi-node deployments.

Other collected metrics remain useful as supporting indicators because they provide additional context about request processing behavior and overall benchmark execution. However, TTFT and TPS are the primary focus of this analysis because they represent the two most important dimensions of inference performance in this validation: how quickly the system begins producing a response, and how much output-token capacity the system can sustain under load. For this reason, the analysis emphasizes TTFT and TPS as the primary comparison points.

The measured values are generally aligned with expected behavior for the AMD Instinct MI300X GPU systems used in the test environment and with common industry expectations for interactive and high-throughput inference workloads. Short-context workloads produced lower TTFT and higher throughput, while larger models and long-context workloads produced higher TTFT and lower throughput due to increased prefill, memory, and compute requirements. This pattern is consistent with expected large language model serving behavior.

The short-context results are favorable for interactive inference because TTFT values were in the few-hundred-millisecond range while output token throughput scaled strongly in the Multi-node path. The long-context results are also directionally positive for document-analysis and RAG-style workloads, but they should be interpreted differently because longer prompts naturally increase prefill work and TTFT. In this validation, the strongest results were observed with Llama 3.1 8B FP16 and with Multi-node throughput scaling across all model profiles.

The results should be interpreted as benchmark characterization results rather than universal model-performance limits. Each test case used a specific combination of model, precision, context length, concurrency, request count, and serving path. Therefore, the most meaningful comparisons are the matched Single-node and Multi-node pairs, the short-context versus long-context behavior within the same model family, and the FP16 versus FP8 behavior for comparable scenarios.

Time to First Token: Single-node SGLang vs. Multi-node Envoy Load balanced Inference

Table 6. Time to First Token Comparison Single-node and Multi-node per model

Scenario Key	Single-node	Multi-node
Llama 3.1 8B FP16, short	190.08	442.04
Llama 3.1 8B FP16, long	286.22	347.73
Llama 3.3 70B FP8, short	302.97	309.99
Llama 3.3 70B FP8, long	307.94	299.63
Llama 3.3 70B FP16, short	303.24	296.62

Llama 3.3 70B FP16, long	1581.2	1627.09
Qwen 2.5 72B FP8, short	449.9	296.85
Qwen 2.5 72B FP8, long	1821.54	887.67
Qwen 2.5 72B FP16, short	355	351.24
Qwen 2.5 72B FP16, long	2767.29	1694.95

Figure 5. Time to First Token by Model: Single-node SGLang vs. Multi-node Envoy

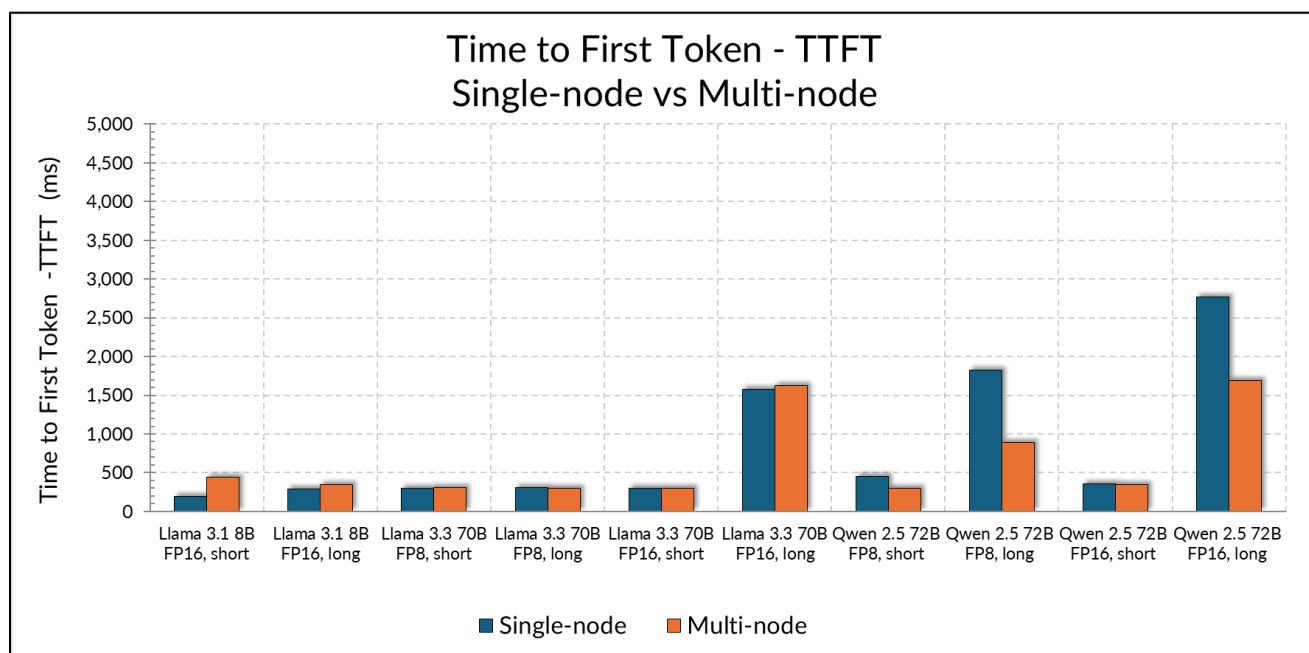


Table 6 and Figure 5 compare Time to First Token across the validated Single-node and Multi-node inference tests.

The TTFT results show that short-context workloads generally delivered the best responsiveness. Most short-context tests completed TTFT in the few-hundred-millisecond range, which is appropriate for interactive inference use cases where users expect the response to begin quickly. Long-context tests showed higher TTFT, especially for the larger Llama 3.3 70B and Qwen 2.5 72B models. This behavior is expected because long-context requests require more input-token processing before the first output token can be generated. In large language model serving, this prefill stage is often one of the main contributors to TTFT.

TTFT values in this report were measured from the benchmark client within the validated test environment. They do not include additional WAN, internet, client-to-cloud, security gateway, or application-layer delay that may be present in a production deployment. In production environments, user-perceived time to first response should account for

both the inference-serving TTFT measured in this validation and any additional delay introduced by the external client access path.

Qwen 2.5 72B and Llama 3.3 70B showed longer TTFT than Llama 3.1 8B because the larger models require more compute, more memory movement, and more scheduling work per request. The difference was especially visible in the long-context scenarios. For example, Qwen 2.5 72B FP16 long-context measured 2767.29 ms TTFT in the Single-node path and 1694.95 ms in the Multi-node path, while Llama 3.3 70B FP16 long-context measured 1581.20 ms and 1627.09 ms respectively. These results reflect the additional prefill and model execution cost associated with larger parameter counts and longer input sequences.

The Multi-node Envoy path improved TTFT in several larger-model scenarios, particularly for Qwen 2.5 72B. Qwen 2.5 72B FP8 long-context improved from 1821.54 ms to 887.67 ms, and Qwen 2.5 72B FP16 long-context improved from 2767.29 ms to 1694.95 ms. These improvements suggest that the Single-node tests were more constrained by backend processing or queueing pressure, while distributing the workload across two MI300X inference servers reduced per-server pressure enough to improve time to first token.

TTFT did not improve in every Multi-node scenario because the Multi-node tests were not identical copies of the Single-node tests with only the server count changed. Several Multi-node scenarios used different concurrency levels and request counts to evaluate scale-out behavior. Higher concurrency can improve aggregate throughput by increasing batching efficiency, but it can also increase queueing before the first token is generated. In addition, the Envoy path adds a forwarding step, and each backend SGLang instance may make different batching and scheduling decisions depending on the active request mix. Therefore, readers should interpret TTFT and TPS together: the Multi-node path consistently improved aggregate throughput, while TTFT varied based on the balance between per-server load reduction, batching behavior, concurrency, and model execution characteristics.

Overall, the TTFT results show that the validated frontend fabric can support both low-latency short-context inference and more demanding long-context workloads. The results also show that model size, context length, precision, concurrency, and request distribution strategy all influence user-visible responsiveness.

Output Token Throughput: Single-node SGLang vs. Multi-node Envoy Load balanced Inference

Table 7. Output Tokens per Second Single-node vs Multi-node comparison per model

Model	Single-node	Multi-node
Llama 3.1 8B FP16, short	85721.93	177829.57
Llama 3.1 8B FP16, long	51727.87	106404.06
Llama 3.3 70B FP8, short	19262.77	33646.61
Llama 3.3 70B FP8, long	8225	17350.18

Llama 3.3 70B FP16, short	20851.89	33846.52
Llama 3.3 70B FP16, long	2389.38	4163.57
Qwen 2.5 72B FP8, short	20302.38	32073.03
Qwen 2.5 72B FP8, long	1862.82	4969.92
Qwen 2.5 72B FP16, short	17400.53	33016.8
Qwen 2.5 72B FP16, long	1701.57	4490.07

Figure 6. Output Token Throughput by Model: Single-node SGLang vs. Multi-node Envoy

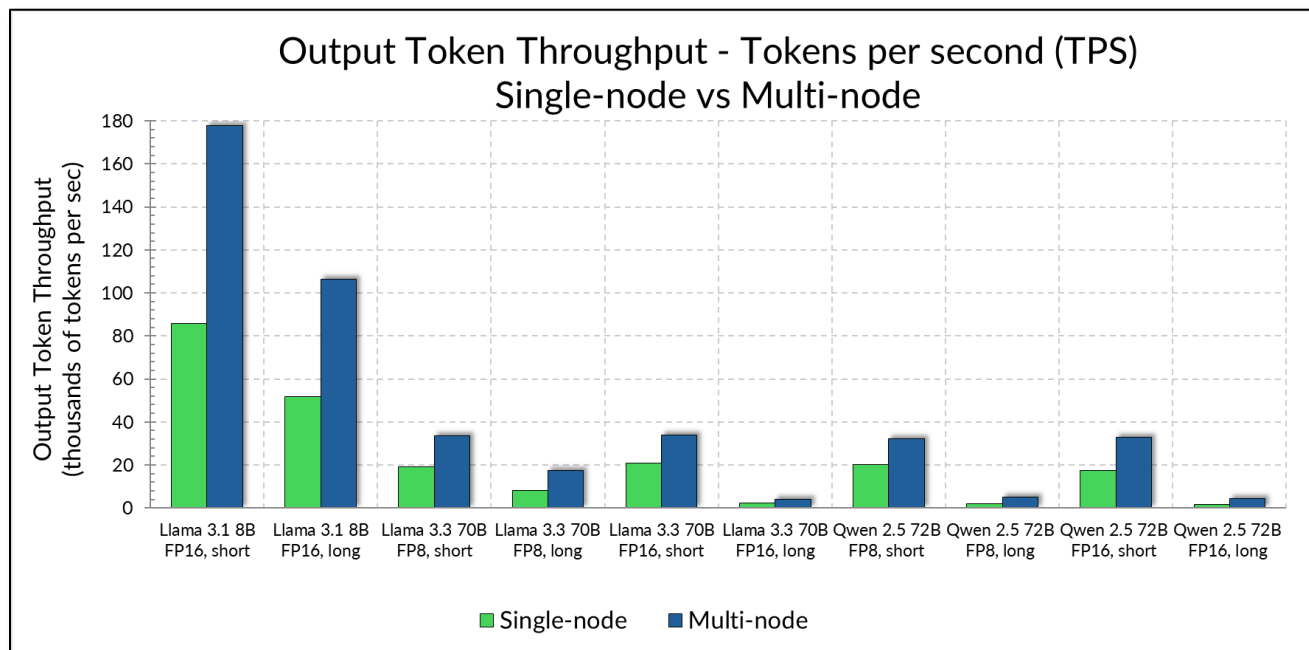


Table 7 and Figure 6 compare output token throughput across the validated Single-node and Multi-node inference tests.

The output TPS results show that Multi-node Envoy load balanced inference increased throughput for every matched model and context profile. This is expected because the Multi-node path distributes requests across two AMD Instinct MI300X servers instead of sending all requests to a single SGLang endpoint. The result validates the expected scale-out behavior of the frontend inference architecture: as additional inference nodes are added behind Envoy, aggregate output token generation capacity increases.

Llama 3.1 8B FP16 delivered the highest output token throughput because it is the smallest model in the test set. Smaller models require less GPU memory, less memory bandwidth, and less compute per generated token than 70B-

class and 72B-class models. As a result, the execution engine can batch more requests efficiently and keep the GPUs highly utilized during the throughput phase. Llama 3.1 8B FP16 short-context increased from 85,721.93 tokens/sec in the Single-node test to 177,829.57 tokens/sec in the Multi-node test, representing approximately 2.07x scaling. This near-linear result is expected for a smaller model that requires less GPU-to-GPU synchronization bandwidth and allows the node to maximize the raw compute potential of the GPU matrix cores.

Short-context tests generally produced higher output TPS than long-context tests. Short-context requests require less input-token processing and typically place less pressure on memory capacity, KV cache usage, and scheduling resources. Long-context workloads, such as RAG-style summarization or document analysis, require more prompt processing and consume more memory resources before and during generation. This reduces the amount of effective batching and lowers aggregate token throughput. For example, Llama 3.1 8B FP16 Single-node throughput was 85,721.93 tokens/sec for short context and 51,727.87 tokens/sec for long context. The same pattern was observed in the Multi-node path, where short-context throughput reached 177,829.57 tokens/sec compared to 106,404.06 tokens/sec for long context.

The results also show that precision affects throughput, but the benefit depends on the model, context profile, and serving configuration. For large models, FP8 can reduce memory pressure and improve effective throughput by lowering the amount of data moved during inference. In the Qwen 2.5 72B short-context Single-node tests, FP16 achieved 17,400.53 tokens/sec at a concurrency of 1,920, while FP8 achieved 20,302.38 tokens/sec at a concurrency of 2,048. This represents an approximately 16.7% throughput improvement, which is consistent with the expected behavior of reduced-precision inference when memory pressure is a limiting factor. However, FP8 did not outperform FP16 in every matched comparison, so precision should be validated per model and workload rather than assumed to provide a universal improvement.

Throughput scaling varied by model family and context profile. The strongest scaling appeared in several long-context Qwen 2.5 72B tests, where the Multi-node path improved output TPS by more than 2.6x. Llama 3.1 8B FP16 also scaled very efficiently, with approximately 2.07x improvement in the short-context test and 2.06x improvement in the long-context test. Other large-model short-context tests showed smaller but still meaningful improvements. This variation reinforces that Multi-node scaling depends on the full serving stack, including model size, precision, context length, concurrency, batching efficiency, and load balancer behavior.

Model Family, Context Length, and Precision Effects

Model family and model size had a clear influence on the measured results. Llama 3.1 8B provided the highest throughput and generally strong responsiveness because it requires significantly less compute and memory per request than the larger 70B and 72B models. Llama 3.3 70B and Qwen 2.5 72B produced lower output TPS and higher TTFT, particularly in long-context scenarios, because they require more GPU resources for prefill, generation, memory movement, and runtime scheduling.

Context length also had a major effect. Short-context workloads are better aligned with low-latency interactive inference because they reduce prefill time and enable more efficient batching. Long-context workloads are more representative of document-analysis, summarization, and RAG-style use cases, where additional input tokens increase TTFT and reduce aggregate throughput.

Precision-format behavior was workload-dependent. FP8 can improve throughput for some large-model scenarios by reducing memory pressure and enabling higher effective concurrency. However, the results show that FP8 and FP16 behavior must be evaluated for each model, context length, and serving configuration. The best precision choice depends on the specific balance between compute efficiency, memory pressure, batching behavior, and model runtime characteristics.

Frontend Network Observations

The validated frontend fabric supported the tested inference traffic paths between the GenAI-Perf client, Envoy load balancer, and SGLang inference servers. The benchmark results show successful request generation, request distribution, response delivery, and metric collection across both Single-node and Multi-node scenarios.

This report does not claim a maximum frontend-fabric utilization limit or conclude that the network could not become a bottleneck under different traffic conditions. A formal network-bottleneck analysis would require correlated fabric telemetry, including link utilization, interface errors, packet drops, retransmissions, queue depth, and congestion indicators during the benchmark runs. Based on the data collected for this validation, the reported performance differences are interpreted primarily through inference-serving factors such as model size, context length, precision, concurrency, batching behavior, and backend request distribution.

Conclusion

The benchmark results demonstrate that the HPE Juniper Networks QFX-based frontend IP fabric can support high-performance AI inference traffic between NVIDIA GenAI-Perf clients, optional Envoy load balancing services, and AMD Instinct MI300X inference servers running SGLang. The validated frontend architecture provided a stable path for both Single-node SGLang inference and Multi-node Envoy load balanced inference. Across the tested scenarios, the network supported inference interactions between clients and servers seamlessly, including request generation, request forwarding, backend inference processing, response delivery, and metric collection.

The results show that Time to First Token and output Tokens per Second are the most meaningful primary metrics for this JVD. TTFT represents the customer-visible delay before a response begins, making it a direct measure of perceived responsiveness for inference applications. TPS represents aggregate output-generation capacity, making it a strong indicator of how efficiently the inference environment serves concurrent demand. Together, these metrics characterize both user experience and scale-out serving efficiency.

The primary conclusion is that the frontend architecture supports scale-out inference behavior. Adding a second AMD Instinct MI300X inference server behind Envoy increased aggregate output token throughput across every validated model and context profile. The Llama 3.1 8B FP16 short-context test showed particularly strong scaling, increasing from 85,721.93 tokens/sec in the Single-node path to 177,829.57 tokens/sec in the Multi-node path, or approximately 2.07x. This near-linear scaling is expected for a smaller model that requires less memory capacity, less memory bandwidth, and less GPU-to-GPU synchronization than the larger 70B and 72B models.

The TTFT results show that short-context workloads generally provided the best responsiveness, while long-context workloads increased TTFT because they require more input-token processing before generation begins. Larger models, including Llama 3.3 70B and Qwen 2.5 72B, showed higher TTFT than Llama 3.1 8B because they require more compute and memory resources per request. This behavior is expected and highlights why model size and context length must be considered when designing and tuning inference services.

Precision also influenced the results. FP8 improved throughput in selected large-model scenarios by reducing memory pressure and enabling higher effective concurrency. For example, Qwen 2.5 72B short-context Single-node throughput improved from 17,400.53 tokens/sec with FP16 to 20,302.38 tokens/sec with FP8, an approximately 16.7% increase. However, FP8 did not universally outperform FP16 across all tests, so precision selection should be validated per model and workload.

Overall, the results support the JVD objective of validating a standards-based Ethernet frontend fabric for AI inference traffic. The QFX5140-24CD8O is validated as a frontend leaf platform for production inference deployments, while HPE Juniper Apstra Data Center Director provides intent-based fabric deployment and operational visibility. The measured results show that the validated frontend fabric can support both responsive Single-node inference and higher-throughput Multi-node inference using Envoy-based request distribution across AMD Instinct MI300X systems.

Appendix A - Test Environment Details

Validated Software Versions

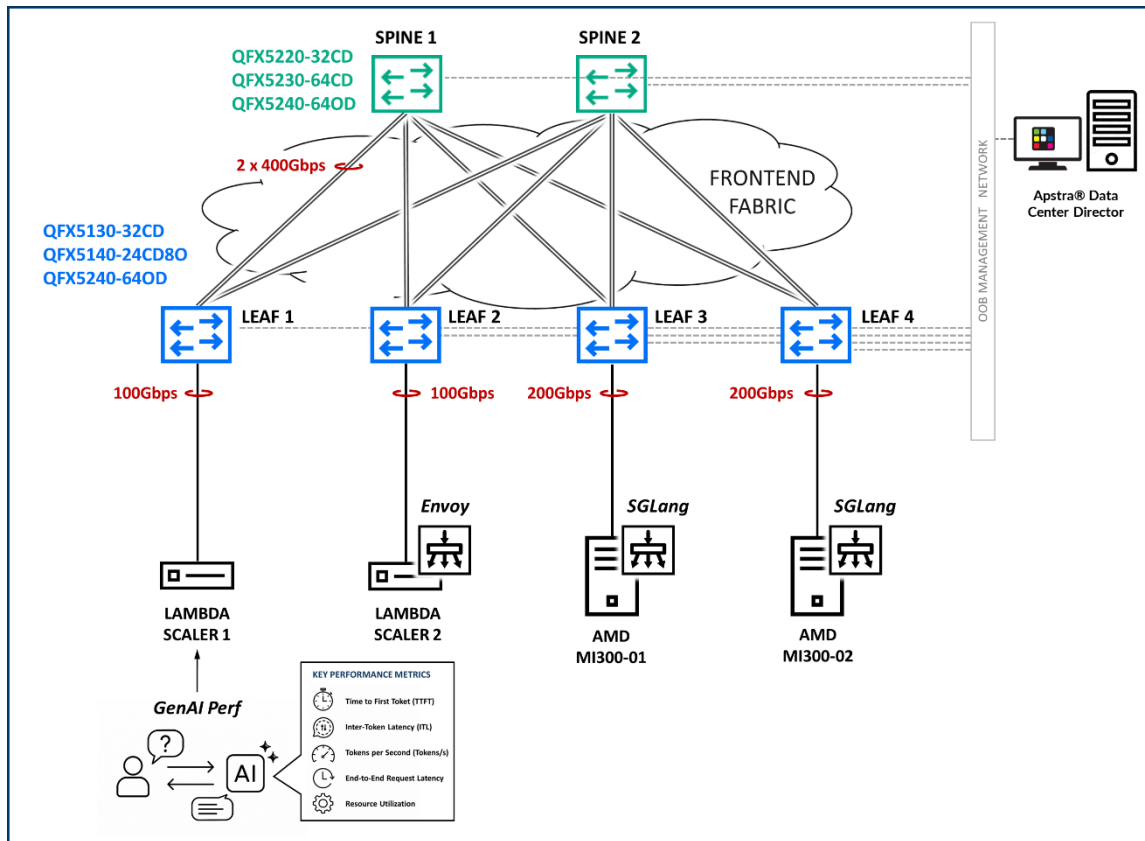
Table A1: Validated Software Versions

Software	Role	Version
QFX devices	Frontend leaf and spine nodes	25.2X100-D20.4-EVO
AMD MI300X server	Inference GPU server	Ubuntu 22.04.5 LTS 6.8.0-111-generic
HPE Juniper Apstra Data Center Director	Fabric automation and operations	6.1
SGLang	Inference serving framework	0.4.5
SGLang Router	Request routing across local GPU-backed workers	0.1.4
Envoy Proxy Service	Load balancing across MI300X inference servers	1.35.3
NVIDIA GenAI-Perf	Inference benchmark load generator	0.0.11
Models	Inference workloads used to validate performance across different model sizes, concurrency levels, and frontend fabric traffic patterns.	Llama 3.1 8B Llama 3.3 70B Qwen 2.5 72B

Appendix B - Test Topology Details

Figure B1 shows the high-level frontend fabric topology used during testing.

Figure B1. AI Inference Frontend Fabric Topology



HPE Juniper Apstra Data Center Director provides intent-based fabric automation for the validated topology. Apstra assigns IP addressing, autonomous system numbers, loopback addressing, and other fabric parameters, then creates and deploys the configuration to each QFX device. It also supports operational consistency by maintaining the intended fabric state and providing visibility into the deployed topology.

The frontend fabric uses eBGP between leaf and spine nodes to provide routed IP reachability, path redundancy, and equal-cost forwarding. This approach allows inference request and response traffic to use redundant fabric paths while keeping the frontend network standards-based and operationally simple.

The topology used to validate the design includes four leaf nodes and two spine nodes.

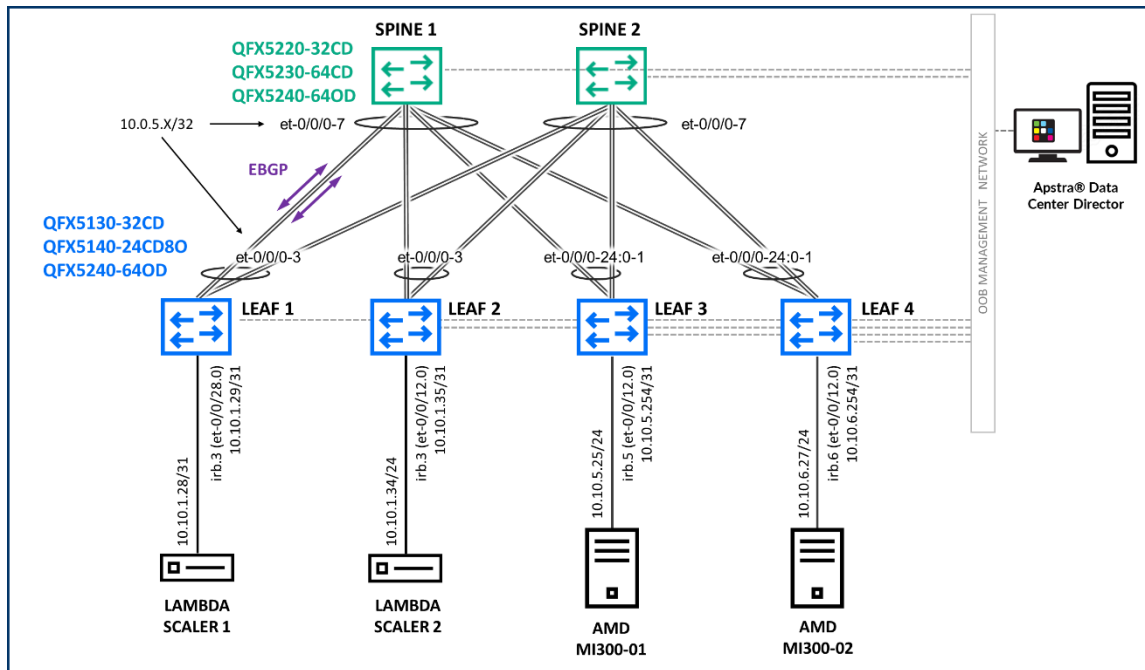
Each frontend leaf connects to both spine nodes using 2 x 400GbE Ethernet links, providing redundant and scalable connectivity across the frontend fabric.

The AMD Instinct MI300X GPU servers connect to the leaf using 400GbE Ethernet links with ConnectX-7 NICs.

The Lambda Scaler systems running Envoy and GenAI-Perf connect to leaf nodes using 100GbE Ethernet links with ConnectX-6 NICs.

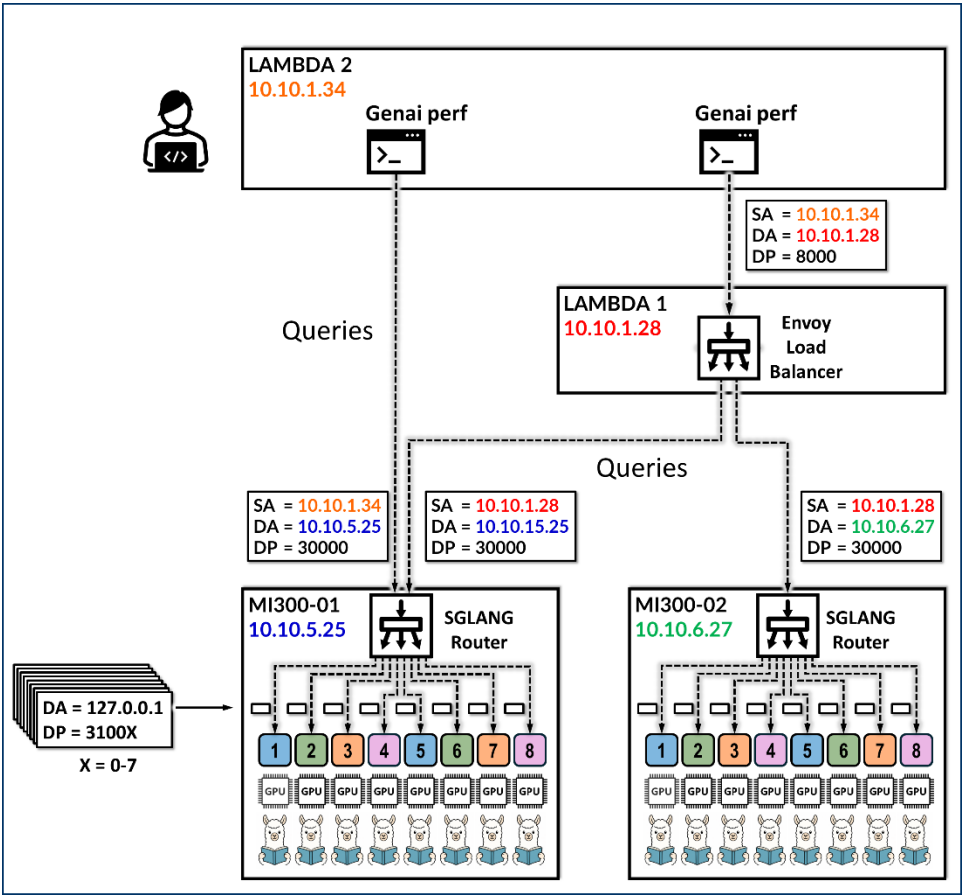
Note: Additional NICs and link speeds may be added in future updates of this JVD.

Figure B2. AI Inference Frontend Fabric Implementation Details



Appendix C - Test Scenario Details

Figure C1. GenAI-Perf, Envoy, and SGLang Inference Traffic Flow and Test Approach



The benchmark plan evaluates two primary performance scenarios using the same naming as the main document:

1. Single-node SGLang inference: GenAI-Perf sends requests directly to the SGLang Router running on one AMD MI300X server to establish a Single-node baseline.
2. Multi-node Envoy load balanced inference: GenAI-Perf sends requests to Envoy, and Envoy distributes the requests across two AMD MI300X servers running SGLang.

Single-node SGLang Inference

In the Single-node SGLang inference scenario, GenAI-Perf sends requests directly to the SGLang Router running on one AMD MI300X inference server. This path bypasses Envoy and is used to establish a Single-node baseline for inference performance.

Table C1: Single-node SGLang Inference Summary

Field	Example / Purpose
Source	GenAI-Perf client host
Example source address	10.10.1.34
Destination	MI300X inference server (MI300-01)
Example destination address	10.10.5.25
Destination port	30000 for SGLang Router service port
Traffic behavior	GenAI-Perf sends requests to the SGLang Router on one inference server.
Purpose	Multi-node baseline and isolated inference endpoint validation.

Multi-node Envoy Load balanced for Inference

In the Multi-node Envoy load balanced scenario, GenAI-Perf sends requests to an Envoy frontend endpoint instead of targeting an MI300X inference server directly.

Envoy distributes requests across the available MI300X inference servers running SGLang. This scenario validates scale-out inference behavior through a frontend service endpoint.

Table C2: Multi-node Envoy Load balanced Inference Summary

Field	Example / Purpose
Source	GenAI-Perf client host
Example source address	10.10.1.34
Frontend endpoint	Envoy load balancer
Example Envoy address	10.10.1.28
Envoy frontend port	8000
Backend endpoints	MI300X inference servers running SGLang
Example server 1	MI300-01, 10.10.5.25, destination port 30000
Example server 2	MI300-02, 10.10.6.27, destination port 30000
Traffic behavior	GenAI-Perf sends requests to Envoy; Envoy forwards requests to inference servers.
Purpose	Scale-out inference validation across multiple inference servers.

SGLang Router and Worker Behavior

Each AMD MI300X server runs one SGLang Router and eight local GPU-backed worker processes. The selected LLM is loaded on each GPU, resulting in one local model instance per GPU.

The SGLang Router listens for inference requests on port 30000 and forwards each request to a local worker. Worker traffic remains local to the MI300X server through loopback interfaces and is not frontend fabric traffic.

Table C3: AI Models Tested - Model Precision

Model Precision	Description
FP16	Higher-precision format used as a baseline for inference testing.
FP8	8-bit floating-point format that can lower memory usage and improve throughput on supported hardware.

Table C4: AI Models Tested - Context Profile

Context Profile	Description
Short	Smaller prompt and response sizes, representative of typical lower-latency inference requests.
Long	Larger prompt and response sizes, such as RAG-style technical document summaries.

Appendix D - Test Methodology Details

This appendix provides the GenAI-Perf command examples, request parameters, metrics, and benchmark scenario matrix used during validation.

Table D1: Examples of GenAI-Perf Commands to Generate Inference Requests

Scenario	Example Command
Single-node SGLang connectivity test	<pre>genai-perf profile -m llama-3.1-8b \ --service-kind openai \ --endpoint-type chat \ --synthetic-input-tokens-mean 100 \ --output-tokens-mean 200 \ --streaming \ --concurrency 10 \ --request-count 20 \ --warmup-request-count 10 \ --url http://10.10.5.25:30000</pre> where 10.10.5.25 is the address of MI300-01
Multi-node Envoy connectivity test	<pre>genai-perf profile -m llama-3.1-8b \ --service-kind openai \ </pre>

	<pre>--endpoint-type chat \ --synthetic-input-tokens-mean 100 \ --output-tokens-mean 100 \ --streaming \ --concurrency 16 \ --request-count 64 \ --warmup-request-count 16 \ --url http://10.10.1.28:8000</pre> where 10.10.1.28 is the address of Envoy.
--	---

Note: The GenAI-Perf command parameter -m identifies the model alias used by the OpenAI-compatible inference endpoint. The model aliases used for this validation are listed in Table D2.

The GenAI-Perf command parameters --synthetic-input-tokens-mean, --output-tokens-mean, --concurrency, --request-count, and --warmup-request-count correspond to ISL, OSL, Concurrency, Request Count, and Warm-up, respectively, as described in Table D3. The values used for each test case are summarized in Table D4.

Model precision, such as FP16 or FP8, is determined by the model checkpoint and serving configuration loaded on the inference server. The context profile, such as short context or long context, is represented by the ISL and OSL values used in the GenAI-Perf command.

Table D2: Model alias for GenAI-Perf

Validation table model -m alias

Llama 3.1 8B	llama-3.1-8b
Llama 3.3 70B	llama-3.3-70b
Qwen 2.5 72B	qwen-2.5-72b

Table D3: GenAI-Perf Inference Request Parameters Definitions

Parameter	Description
Concurrency	Number of simultaneous inference requests or active request streams generated during the benchmark test. Increasing concurrency helps evaluate how the inference service behaves under higher demand.
Number of Requests	Total number of inference requests generated during the benchmark run. This value determines the size of the test sample used to calculate performance metrics.

Input Sequence Length (ISL)	Measured in tokens. Number of input tokens submitted with each request. (Prompt length).
Output Sequence Length (OSL)	Measured in tokens. Number of output tokens generated in each response.
Warm-up	Initial benchmark period or set of requests used to prepare the inference service before collecting final performance results. Warm-up helps reduce the effect of startup behavior, model initialization, caching, and other transient conditions.

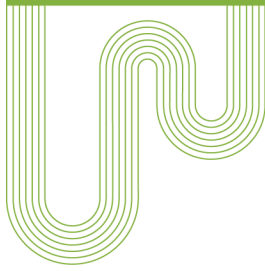
Table D4: Benchmark Scenario Matrix

Test ID	Path	Model	Concurrency	Request Count	ISL	OSL	Warm-up
T1	Single-node	Llama 3.1 8B FP16, short context	2624	5248	100	100	2624
T2	Multi-node	Llama 3.1 8B FP16, short context	5888	6000	100	100	5888
T3	Single-node	Llama 3.1 8B FP16, long context	1200	2400	1500	200	1200
T4	Multi-node	Llama 3.1 8B FP16, long context	2700	5400	1500	200	2700
T5	Single-node	Llama 3.3 70B FP8, short context	1500	3000	100	100	1500
T6	Multi-node	Llama 3.3 70B FP8, short context	2500	5000	100	100	2500
T7	Single-node	Llama 3.3 70B FP8, long context	500	1000	1500	1200	500
T8	Multi-node	Llama 3.3 70B FP8, long context	1100	2200	1500	1200	1100
T9	Single-node	Llama 3.3 70B FP16, short context	2000	4000	100	100	2000
T10	Multi-node	Llama 3.3 70B FP16, short context	3000	6000	100	100	3000
T11	Single-node	Llama 3.3 70B FP16, long context	256	384	1500	200	256
T12	Multi-node	Llama 3.3 70B FP16, long context	320	640	1500	200	320
T13	Single-node	Qwen 2.5 72B FP16, short context	1920	3840	100	100	1920
T14	Multi-node	Qwen 2.5 72B FP16, short context	2560	5120	100	100	2560
T15	Single-node	Qwen 2.5 72B FP16, long context	128	256	1500	200	128
T16	Multi-node	Qwen 2.5 72B FP16, long context	256	512	1500	200	256
T17	Single-node	Qwen 2.5 72B FP8, short context	2048	4096	100	100	2048
T18	Multi-node	Qwen 2.5 72B FP8, short context	2624	5248	100	100	2624
T19	Single-node	Qwen 2.5 72B FP8, long context	128	256	1500	200	128
T20	Multi-node	Qwen 2.5 72B FP8, long context	256	512	1500	200	256

Note: In Table D4, Single-node refers to the direct SGLang test path using one AMD Instinct MI300X server from the validated lab topology. Multi-node refers to the Envoy load balanced test path using both AMD Instinct MI300X servers in the validated lab topology.

Table D5: Inference Performance Metrics

Metric Category	Metric	Description
Responsiveness	Time to First Token (TTFT)	Measured in milliseconds. Elapsed time between request submission and the first generated token. Indicates perceived responsiveness and interactive user experience.
Responsiveness	Time to First Output (TTFO)	Measured in milliseconds. Elapsed time between request submission and the first output received by the client. For streaming text generation, this is often closely related to TTFT.
Request Latency	Request Latency	Measured in milliseconds. Total elapsed time from request submission to completed response. Captures full user-facing response time.
Streaming Performance	Time to Second Token (TTST)	Measured in milliseconds. Elapsed time between request submission and the second generated token. Helps characterize early streaming behavior after the first token is returned.
Streaming Performance	Inter-Token Latency (ITL)	Measured in milliseconds. Delay between generated output tokens during response streaming. Indicates response smoothness and token generation consistency.
Throughput	Output Tokens per Second (TPS)	Measured in tokens per second. Rate of output token generation by the system. Measures aggregate inference throughput and serving efficiency.
Throughput	Output Tokens per Second per User	Measured in tokens per second per user. Average output token generation rate experienced by each concurrent user or request stream.
Throughput	Request Throughput	Measured in requests per second. Number of inference requests completed by the system per second. Indicates how many requests the service can handle under benchmark load.

**Corporate and Sales Headquarters**

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089 USA
Phone: 888.JUNIPER (888.586.4737)
or +1.408.745.2000
Fax: +1.408.745.2100
www.juniper.net

APAC and EMEA Headquarters

Juniper Networks International B.V.
Boeing Avenue 240
1119 PZ Schiphol-Rijk
Amsterdam, The Netherlands
Phone: +31.207.125.700
Fax: +31.207.125.701

Copyright 2025 Juniper Networks, Inc. All rights reserved. Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. in the United States and other countries. All other trademarks, service marks, registered marks, or registered service marks are the property of their respective owners. Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Send feedback to: design-center-comments@juniper.net V1/260610