Confidential Computing on nVIDIA H100 GPU: A Performance Benchmark Study

Jianwei Zhu
jianweiz@phala.network

Hang Yin
hangyin@phala.network

Shunfan Zhou
shelvenzhou@phala.network

September 6, 2024

Abstract

This report evaluates the performance impact of enabling Trusted Execution Environments (TEE) on NVIDIA H100 GPUs for large language model (LLM) inference tasks. We benchmark the overhead introduced by TEE mode across various models and token lengths, focusing on the bottleneck caused by CPU-GPU data transfers via PCIe. Our results show that while there is minimal computational overhead within the GPU, the overall performance penalty is primarily due to data transfer. For most typical LLM queries, the overhead remains below 5%, with larger models and longer sequences experiencing near-zero overhead.

1 Introduction

Trusted Execution Environments (TEEs), are increasingly important in machine learning and AI due to rising security requirements across both enterprise and decentralized applications. The introduction of TEE-enabled GPUs, such as the NVIDIA H100, adds an extra layer of protection for sensitive data but may affect performance. Understanding these trade-offs, especially for large-scale machine learning tasks, is crucial for adopting TEE in high-performance AI applications.

This report quantifies the performance overhead of enabling TEE on the NVIDIA H100 GPU during LLM inference tasks, examining where the overhead arises and under what conditions it can be minimized.

2 Background

2.1 Trusted Execution Environment

A TEE is a hardware-based security feature that isolates computations, preventing unauthorized access and tampering—even from the operating system or the physical hardware owner. As the core technology to enable Confidential Computing, TEEs create secure enclaves where sensitive data and code are processed with encryption, ensuring confidentiality and integrity even if the broader system is compromised. Traditionally implemented in CPUs, TEE technology was extended to GPUs by NVIDIA in 2023, enabling tamper-proof and confidentiality-preserving computation inside the GPU with minimal performance penalty.

2.2 nVIDIA H100 GPU

The NVIDIA H100 GPU represents a significant milestone as the first GPU to support TEE. In TEE mode, the H100 operates in an isolated and secure environment where data transfers between the CPU and GPU are encrypted. This is facilitated by "bounce buffers," which protect all inputs and outputs during transit between the CPU's encrypted memory and the GPU's internal memory.

To maintain end-to-end security, the H100 works in conjunction with CPU TEEs, such as Intel's TDX or AMD's SEV-SNP, securing communication channels between the GPU driver and interacting software. This prevents unauthorized access and ensures data integrity throughout the process.

The H100 also implements remote attestation to verify the GPU's identity and the authenticity of its firmware. Secure Boot further strengthens security by ensuring that only authenticated firmware is executed during the GPU's boot process.

2.3 Performance Impact

Enabling TEE on the NVIDIA H100 GPU introduces performance overheads primarily due to additional encryption and decryption during secure data transfer. While the GPU's internal computation remains unaffected, the main bottleneck lies in the CPU-GPU IO, particularly when data is exchanged via PCIe. This impact varies with the size of the data transfer. The following sections present experimental results quantifying these effects across various use cases.

With the TEE-enabled NVIDIA H100 GPU, it becomes crucial to quantify performance tradeoffs during practical use cases. In the next section, we outline the methodology used to assess the performance impact of TEE during LLM inference tasks.

3 Methodology

To evaluate the performance overhead, we conducted experiments comparing inference throughput and latency with TEE mode enabled and disabled, under different models, input and output lengths, and batch size setups. Our primary focus was to reveal the performance penalty in real-world large language model (LLM) inference tasks.

3.1 Metrics

The primary metrics are evaluated following typical evaluation frameworks [AAK+24]:

- **TTFT (Time To First Token):** The time from request arrival to the generation of the first output token. It includes scheduling delay and prompt processing. Lower TTFT is essential for real-time applications, while higher TTFT is tolerable in batch processing.
- **ITL (Inter Token Latency):** The time between generating each token during decoding. This directly affects the perceived model speed. A rate of around 6 tokens per second is necessary for smooth user experience, assuming average reading speed.
- **TPS (Token per Second):** The average rate of token generation during decoding. It's calculated as the total decoding time divided by the number of tokens generated.
- **Latency:** The total execution time per request, including scheduling, prompt processing, and token generation. Lower normalized latency improves system throughput, especially under high query loads.
- **QPS (Query per Second):** The maximum load a system can handle while meeting latency targets. Higher QPS reduces serving costs and is a key measure of system capacity.

3.2 Test Scenarios

Experiments were structured to explore TEE mode impact under diverse conditions:

- **TEE mode ON vs. TEE mode OFF**: Tests were performed with TEE mode alternately enabled and disabled on the H100 GPU, allowing for a direct comparison of performance.
- **Sequence Lengths**: We tested various token lengths by sampling the ShareGPT Dataset [ano] to simulate different LLM inference tasks.
- **Batch Size**: We tested both the fixed batch sizes (1, 4, and 16) and dynamic batch size determined by vLLM [KLZ+23] to simulate the performance for serving real-time requests and batch requests.

3.3 Experimental Setup

3.3.1 Infrastructure

We set up the experiments with the following hardware:

- **GPU**: NVIDIA H100 NVL (94GB, 3.9TB/s bandwidth)
- **CPU**: AMD EPYC 9V84 96-Core Processor with SEV-SNP
- **Memory**: 314 GB
- **Driver versions**:
  - CUDA 12.5 (driver version 555.42.02)
  - Kernel driver version 550.90.07

3.3.2 Application

The experiments utilized the Benchmark suites of `vLLM v0.5.4 (rev: 4db5176)` [KLZ+23].

3.3.3 Models

Three LLMs were used for inference:

- **Meta-Llama-3.1-8B-Instruct**
- **Phi-3-14B-128k-Instruct**
- **Meta-Llama-3.1-70B-Instruct** with 4-bit bitsandbytes quantization to fit into a single H100 GPU

4 Results

Conclusion 1: The average overhead is less than 7%. We quantified the overhead by measuring the throughput with TEE mode enabled versus disabled, across varying input sizes and model configurations as shown in Table 1.

Model	**TPS (tokens/s)**			**QPS (req/s)**		
	TEE-on	**TEE-off**	**Overhead**	**TEE-on**	**TEE-off**	**Overhead**
LLama-3.1-8B	123.2985	132.3618	6.85%	18.2141	18.8208	3.22%
Phi3-14B-128k	66.5845	69.7787	4.58%	7.1760	7.3456	2.31%
Llama-3.1-70B	2.4822	2.4789	-0.13%[1]	0.8325	0.8295	-0.36%[2]

Table 1: Performance comparison of TEE-on and TEE-off modes for various models in terms of TPS (tokens per second) and QPS (queries per second).

The throughput is measured in two ways: the average throughput of the outputted tokens per second (TPS), and that of the parallel requests the hardware can handle (PQS). TPS is measured by running the model with a batch size of 1. It shows the pure latency overhead introduced by the TEE mode. It reflects the performance of realtime requests. QPS is measured by maximizing the query throughput with dynamically optimized batch size. It reflects the minimal average overhead the TEE mode brings.
Conclusion 2: The overhead reduces toward zero as the model size grows. As shown in Table 1, the smallest model (**LLama-3.1-8B**) has the highest overhead. The medium size model (**Phi3-14B-128k**) has roughly 2/3 of the overhead compared with the smaller one. The largest model (**Llama-3.1-70B**) has a trivial overhead close to zero.

[1]The overhead is negative due to the precision loss.
[2]The overhead is negative due to the precision loss.

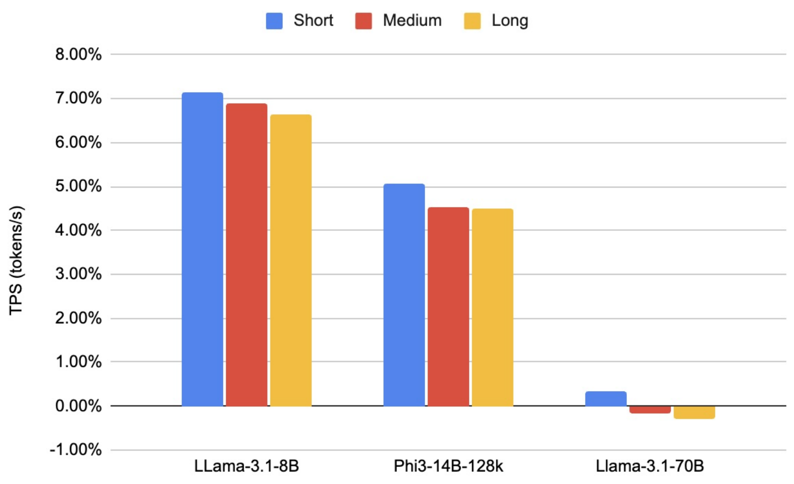

Figure 1: Throughput overhead across different token sizes (length of the input and output sequence). Short sequences are no longer than 100 tokens. Medium sequences are no longer than 500 tokens. Long sequences are between 501 and 1500 tokens.

Conclusion 3: The latency is the main factor contributing to the overhead of the TEE model. Table 2 shows the overhead introduced to the latency measured by TTFT and ITL. TTFT has a higher overhead compared with ITL, indicating the bottleneck is likely introduced by the IO instead of the computation happening inside the TEE. Nevertheless, the overhead becomes trivial when hosting heavy computation models like **Llama-3.1-70B**.

Model	**TTFT (s)**			**ITL (s)**		
	TEE-on	**TEE-off**	**Overhead**	**TEE-on**	**TEE-off**	**Overhead**
LLama-3.1-8B	0.0288	0.0242	19.03%	1.6743	1.5549	7.67%
Phi3-14B-128k	0.0546	0.0463	18.02%	3.7676	3.5784	5.29%
Llama-3.1-70B	0.5108	0.5129	-0.41%[3]	94.8714	95.2395	-0.39%[4]

Table 2: Comparison of TTFT (Time to First Token) and ITL (Inter Output Token Latency) for TEE-on and TEE-off modes across models.

Conclusion 4: The overhead reduces as the token size grows. As shown in Figure 1, the throughput overhead reduces when the sequence length grows, measured by the total input and output token count. The detailed throughput metrics across various sequence length can be found at Table 3.

Model	**TPS - short (tokens/s)**			**TPS - medium (tokens/s)**			**TPS - long (tokens/s)**		
	TEE-on	**TEE-off**	**Overhead**	**TEE-on**	**TEE-off**	**Overhead**	**TEE-on**	**TEE-off**	**Overhead**
LLama-3.1-8B	127.0310	136.8282	7.16%	122.9356	132.0464	6.90%	122.9705	131.7333	6.65%
Phi3-14B-128k	70.9799	74.7556	5.05%	66.1690	69.3104	4.53%	66.2987	69.4176	4.49%
Llama-3.1-70B	2.5983	2.6073	0.34%	2.4413	2.4374	-0.16%[5]	2.5245	2.5168	-0.30%[6]

Table 3: Performance comparison of TEE-on and TEE-off modes across different sequence lengths in terms of TPS (tokens per second). Short sequences are no longer than 100 tokens. Medium sequences are no longer than 500 tokens. Long sequences are between 501 and 1500 tokens.

Conclusion 5: TEE can reach a typical throughput

Our experiments revealed that, with medium-sized inputs, the NVIDIA H100 GPU achieves 130 TPS for Llama3-8B, while the larger Phi-3-14B model reaches approximately 6 TPS. These results

[3]The overhead is negative due to the precision loss.
[4]The overhead is negative due to the precision loss.
[5]The overhead is negative due to the precision loss.
[6]The overhead is negative due to the precision loss.

demonstrate the robust performance of the H100 GPU across models of varying complexity.

More detailed experimental data is shown in Figure 2, 3, and 4.

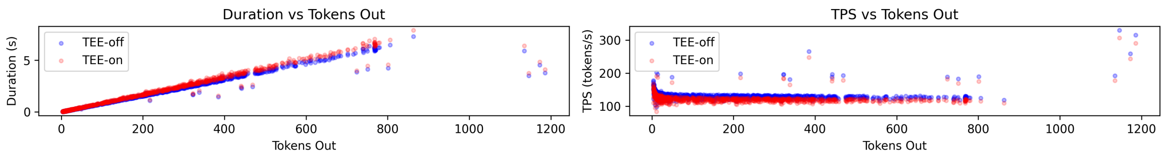

Figure 2: Throughput vs output token size for **LLama-3.1-8B**

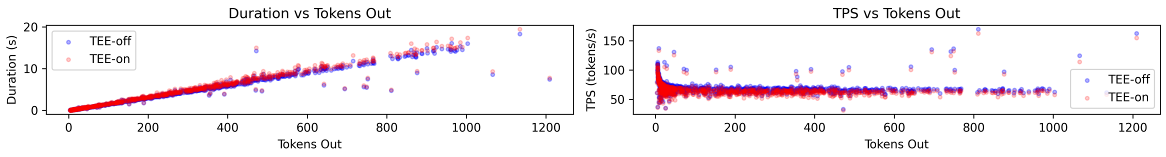

Figure 3: Throughput vs output token size for **Phi3-14B-128k**

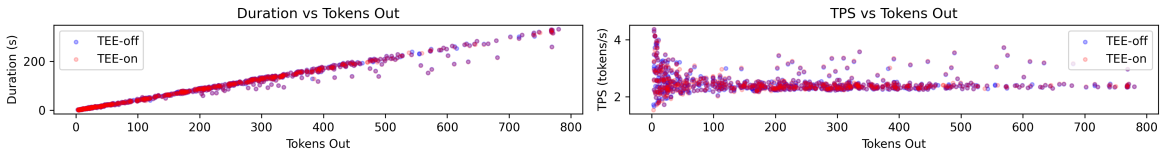

Figure 4: Throughput vs output token size for **Llama-3.1-70B**

5 Conclusion

Our results show that as input size grows, the efficiency of TEE mode increases significantly. When computation time within the GPU dominates overall processing time, the IO overhead introduced by TEE mode diminishes, allowing efficiency to approach nearly 99%.

Efficiency growth is more pronounced in larger models, such as **Phi3-14B-128k** and **Llama-3.1-70B**, due to their greater computational demands, which result in longer GPU processing times. Consequently, the IO overhead becomes increasingly trivial as model size increases.

The total token size (sum of input and output token size) significantly influences the throughput overhead. Larger total token counts lead to higher efficiencies, as they enhance the ratio of computation time to IO time.

These findings underscore the scalability of TEE mode in handling large-scale LLM inference tasks, particularly as input sizes and model complexities grow. The minimal overhead in high-computation scenarios validates its applicability in secure, high-performance AI workloads.

References

[AAK+24] Amey Agrawal, Anmol Agarwal, Nitin Kedia, Jayashree Mohan, Souvik Kundu, Nipun Kwatra, Ramachandran Ramjee, and Alexey Tumanov. Metron: Holistic performance evaluation framework for llm inference systems. *arXiv preprint arXiv:2407.07000*, 2024.

[ano] anon8231489123. Sharegpt vicuna unfiltered. https://huggingface.co/datasets/anon8231489123/ShareGPT_Vicuna_unfiltered. Accessed: 2024-09-04.

[KLZ+23] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.