Confidential Computing for Privacy-Preserving Language Model Deployment

DISSERTATION

Submitted in partial fulfillment of the requirement of the
MTech in Artificial Intelligence and Machine Learning Program

By

AJAY KUMAR PATEL
BITS ID No.: 2023AA05956

Under the supervision of

Mr. Chandan Sharma,
Senior Software Engineering,
Fujitsu Research of India, Bangalore

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
Pilani (Rajasthan) INDIA

June 2025

Acknowledgements

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without mentioning the people who made it possible, because success is the epitome of hard work, perseverance, undeterred missionary zeal, steadfast determination, and most of all “ENCOURAGING GUIDANCE”.

I express my gratitude to my supervisor Mr. Abhishek Jain for providing me with a means of attaining my most cherished goals. I record my heart full of thanks and gratitude to my additional examiner Prof. Vishali and BITS WILP for providing me an opportunity to carry this project, along with purposeful guidance and moral support extended to me throughout the duration of the project work.

I would like to express my love and gratitude to my beloved family, my friends, and my Fujitsu Research team members, for their understanding & motivation throughout the duration of this project.

Ajay Kumar Patel

Abstract

These days, language models are popping up in all sorts of apps, whether it’s chatbots, text generators, or tools that help summarize information. Since they usually need a lot of computing power, most of them run in the cloud. That works well in terms of performance, but it also means there’s always a question mark around how private the data really is, especially if it includes something sensitive. There is a risk that user inputs or internal model data could be accessed by cloud operators or others with elevated system access.

In this project, I plan to explore how Confidential Computing, a technique that protects data even while it is being processed, can be used to safely deploy LLMs on the cloud. For my project, I’ll be working with some of the newer hardware-based security features, which are AMD SEV-SNP, Intel TDX, and arm CCA. These basically allow you to spin up a virtual machine where everything happening inside is kept hidden, even from the host system itself. That way, both the language model and whatever data it’s working with stay protected the whole time.

The goal is to set up a small language model inside a confidential VM and evaluate how well this setup works in practice. I’ll measure how much performance is affected and whether the privacy gains are worth the trade-off. The outcome should help show whether these security techniques are suitable for real-world use in AI-based services.

List of Figures -

List of Tables -

Table of Contents

Introduction

Background

As AI systems increasingly leverage cloud-based Large Language Models (LLMs), for example, chatbots and summarization tools, there is a growing concern about data exposure. While cloud platforms use encryption for data in transit and at rest, runtime data and model state remain vulnerable to inspection by hypervisors or cloud administrators. This exposure poses privacy risks, particularly in sensitive domains such as healthcare and finance.

Problem Statement

The key objective of this work is to show how confidential computing can be used to securely deploy LLM model for inferencing and what is performance difference between hosting a LLM model on confidential environment vs a non-confidential environment.

My current Proof of Concept (POC) serves a lightweight LLM inside a Trusted Execution Environment (TEE), specifically, an AMD SEV-SNP protected VM on Google Cloud. Remote Attestation has been used to demonstrate confidentiality of the machine which guarantee security of data while in use. Remote Attestation provide trust to users that their data and model are shielded from unauthorized host access.

LLM model on Confidential Computing Environment

Language Models

Large Language Models (LLMs) are neural networks trained on massive text corpora through self-supervised learning. They excel in tasks like text generation, summarization, translation, and question answering. These models typically follow a transformer architecture, introduced in "Attention Is All You Need," [1] which uses self-attention mechanisms to process input sequences in parallel, allowing efficient scaling to billions or even trillions of parameters.

Based on language task there are mainly three main architectural variants of Transformer models [2] –

1. Encoder models
2. Decoder models
3. Encoder – Decoder Models (Sequence-to-Sequence Models)

Most modern LLMs use a decoder-only transformer design. This architecture is pretrained to predict the next token in a sequence and can be finetuned for specific tasks, including few-shot or zero-shot learning. The parameter count in production LLMs varies widely, from tens of millions to several hundred billion, impacting their capabilities and resource demands [2].

For our Proof-of-Concept (PoC), we employ DistilGPT2, a compact variant of GPT-2. With approximately 82 million parameters (vs. 124 million in GPT-2), DistilGPT2 maintains strong generation performance while significantly reducing model size and inference time. It was trained via knowledge distillation from GPT-2 using the **OpenWebTextCorpus** dataset.

By choosing DistilGPT2, our PoC strikes a practical balance, it enables experimentation on confidential infrastructures (like AMD SEV-SNP) without extensive computational resources. This setup provides a realistic basis to investigate privacy-preserving model deployment while retaining the core characteristics of larger transformer-based LLMs.

Confidential Computing

Confidential Computing is the protection of data in use by performing computation in a hardware-based, attested Trusted Execution Environment [3]. In cloud environment, host hypervisor is responsible to creation of VMs (Virtual Machine) and managing memory used by these VMs. Any vulnerability on host system can expose the data processed by virtual machine. Confidential Computing solves this issue by removing host environment including hypervisor from Virtual Machine's trust boundary and now even if hypervisor gets compromised, VM's data remain safe.

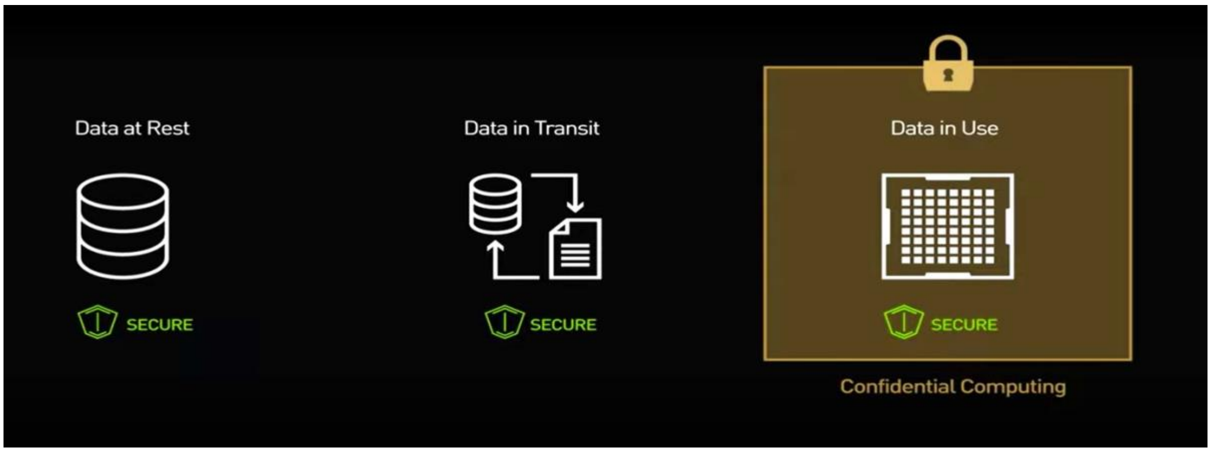


Figure 1 Confidential computing protects data in use while running on a processor

All major CPU vendors, AMD, Intel, Arm, and RISC-V, are actively working on hardware support for Confidential Computing. Below is a summary of the current state:

1. AMD & Intel
    a. AMD and Intel Confidential Computing implementations target primarily high-end processors, typically used in data centres and cloud computing [4].
    b. AMD SEV-SNP support is available since 2021 in 3rd-gen EPYC (Milan) processors, enabling Virtual Machine-level memory encryption and integrity protection.
    c. Intel Introduced TDX support in 2023 with 4th-gen Xeon Scalable CPUs, offering robust VM isolation and attestation.
2. ARM
    a. Arm Confidential Compute Architecture (CCA), announced in 2021 with ARMv9-A, leverages Realms to create shielded execution environments [5].
    b. ARM's implantation applies to both desktop and embedded processors.

    c. Widespread hardware support is in early stages, with many implementations still in simulator or prototype phases.

3. RISC – V
    a. The Confidential VM extension for RISC-V, known as CoVE, is a set of specifications and proposals aimed at enabling confidential computing on RISC-V platforms [6].
    b. Open-source framework, Keystone, enables confidential computing on an existing RISC-V hardware [7].

AMD SEV SNP

In 2016, AMD introduced Secure Encrypted Virtualization (SEV), the x86 technology designed to isolate virtual machines (VMs) from hypervisor. With SEV, VMs could be assigned an unique AES encryption key that is used to automatically encrypt their in-memory data. When components such as hypervisor try to read memory inside a guest, it is only able to see encrypted bytes. In 2017, SEV-ES (Encrypted State) feature was introduced which protects CPU register state. [8]

AMD SEV-SNP (Secure Nested Paging) is next generation of SEV which include security features of SEV and SEV-ES while adding new hardware based security protection. SEV-SNP adds strong memory integrity protection to prevent malicious hypervisor based attacks like data replay, memory remapping, and more in order to create an isolated execution environment. [8]

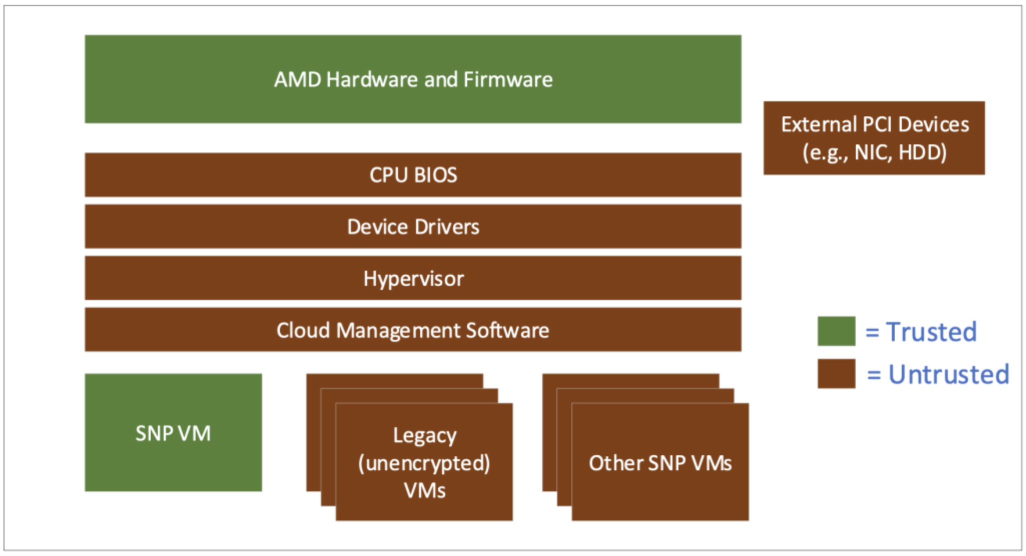


Figure 2 AMD SEV-SNP Thread model

Attestation

A critical aspect of TEEs are the requirements of hardware-based isolation, and attestation (cryptographic-verification) of the Trusted Computing Base (TCB). Attestation verifies the platform on which the VM is running and VM itself. When platform is booted, all the firmware, software components and memory region are measured, and their measurement is stored securely in protected memory. When a VM boots up, attestation agent running inside VM can request for platform and VM measurements which is provided in the form of encrypted report. This report can then be verified by a remote third party, known as Remote Verifier, which based on the report content and verification status, can provide if VM is trusted or not. This process of verifying VM generated report by remote verifier is called **Remote Attestation.**

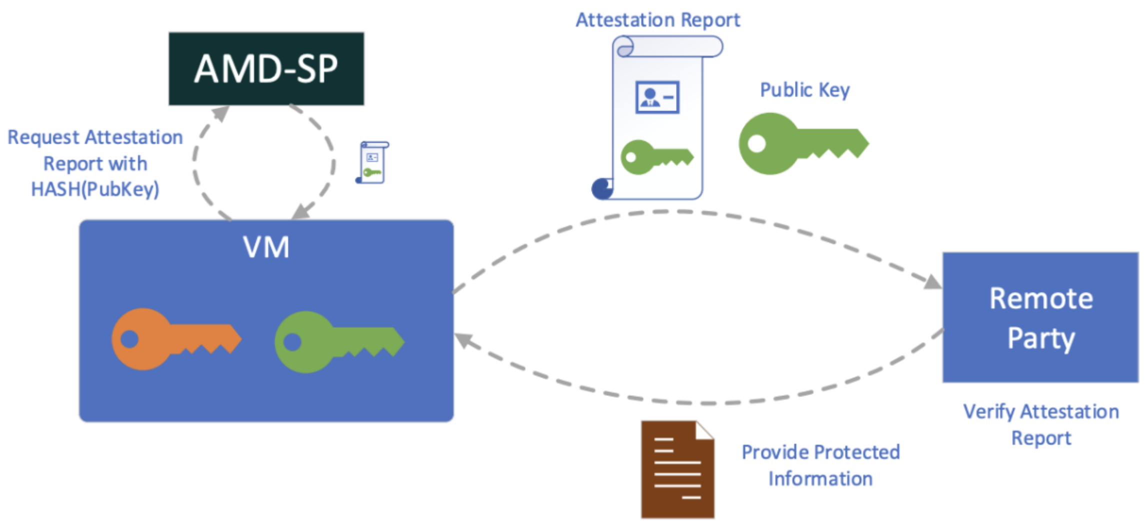

AMD-SP
Request Attestation Report with HASH(PubKey)
VM
Attestation Report
Public Key
Remote Party
Verify Attestation Report
Provide Protected Information

Figure 3 AMD SEV-SNP Attestation

Environment Setup

AMD SEV-SNP VM on Google Cloud

Currently AMD SEV-SNP enabled hardware are available with different cloud provider only in limited regions.

The environment for this Proof of Concept was built using the following stack:

- **Cloud Platform**: Google Cloud Platform (GCP)
- **VM Configuration**:
    - Machine type: n2d-standard-32 (32 vCPUs, 128 GB Memory)
    - CPU platform: AMD Milan (SEV-SNP capable)
    - Architecture: x86/64
- **Language Model Runtime**:
    - Model: distilgpt2 from Hugging Face Transformers [9]
    - Web Framework: FastAPI
    - Language: Python
- **Attestation Tool**:
    - virtee/snpguest [10]

This setup enables us to securely serve an LLM from within a hardware-protected environment while enforcing remote attestation before allowing sensitive user interaction.

Proof of Concept Design and Working

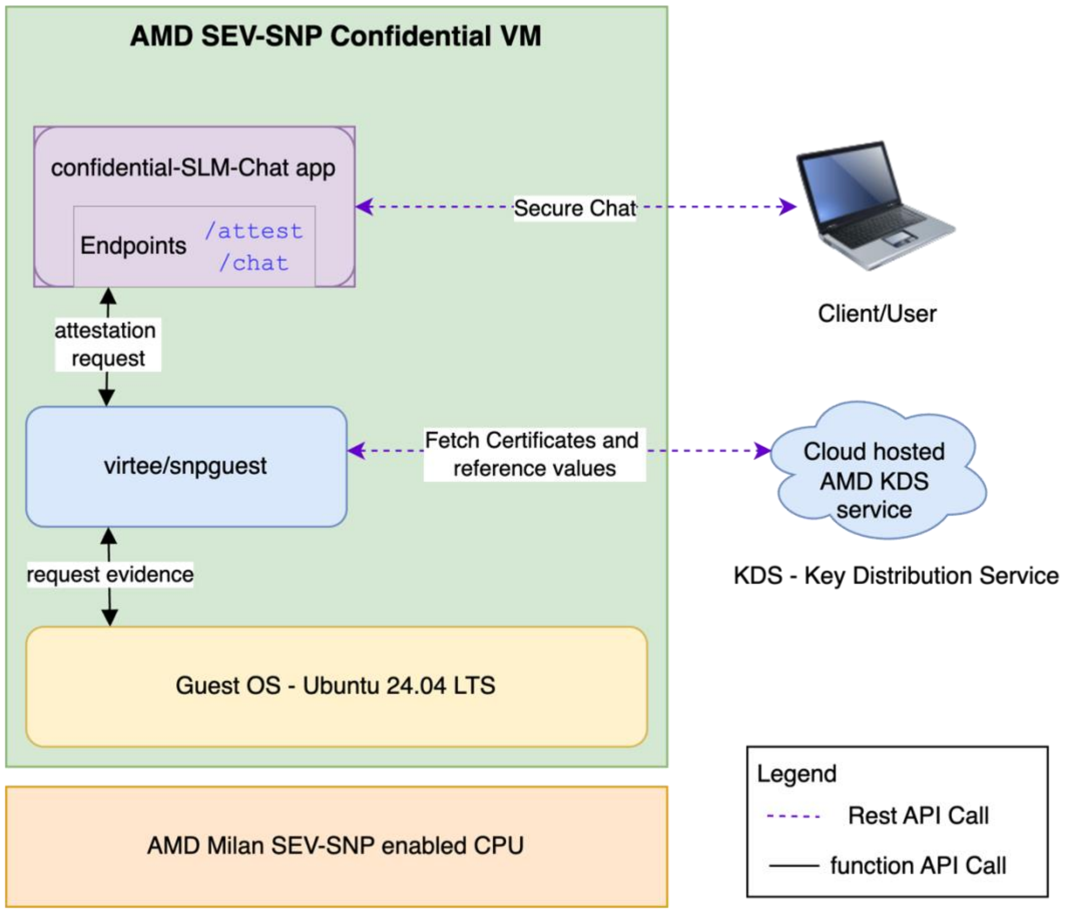

AMD SEV-SNP Confidential VM
confidential-SLM-Chat app
Endpoints
/attest
/chat
Secure Chat
Client/User
attestation request
virtee/snpguest
Fetch Certificates and reference values
Cloud hosted AMD KDS service
KDS - Key Distribution Service
request evidence
Guest OS - Ubuntu 24.04 LTS
AMD Milan SEV-SNP enabled CPU
Legend
Rest API Call
function API Call

Figure 4 Confidential-SLM-chat PoC app design

PoC app, named Confidential-SLM-Chat, is designed to demonstrate privacy preserving secure interaction with language model, where model and chat processed on the confidential VM is protected and shielded from host environment (cloud provider, GCP in this case).

Different entities involved in above architecture are -

1. **Client** – End user interacting with chat application using web UI.

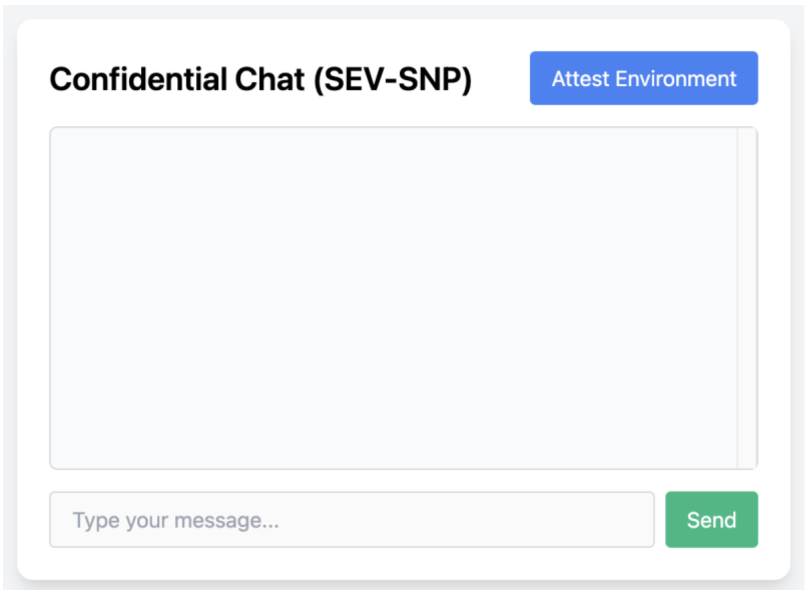
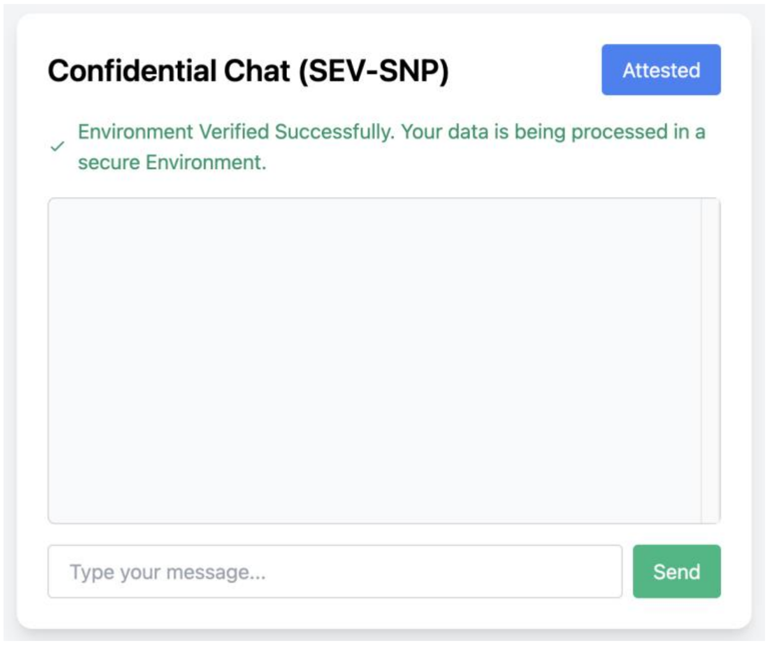


Figure 5 Client Application initial UI before attestation



✓ Environment Verified Successfully. Your data is being processed in a secure Environment.

Figure 6 Client Application UI after successful attestation

2. **Confidential-SLM-Chat server** – Application server built using FastAPI python framework. Provides two endpoints for user –
    a. "/attest" – This rest endpoint when called perform attestation (verification of instance) and return status as "verified" or "not verified".
    b. "/chat" – This endpoint is used to interact with language model and language model response.

If attestation result of VM is "not verified" then user can get to know that VM is compromised and user data is not secure inside VM. User can avoid sending its data to language model.

3. **Virtee/snpguest** – A CLI tool for interacting with SEV-SNP guest environment. It has utilities to generate and verify attestation report, which in turn provide trustworthiness of VM.
4. **KDS (AMD Key Distribution Service)** – This is a cloud service hosted by AMD. It provides root certificates for verification of attestation report signature and reference values for the appraisal of evidence in report.

POC GitHub URL - https://github.com/patelajaychh/confidential-slm-chat

Benchmarking and Performance Evaluation

This section presents a comparative analysis of running selected small language models (SLMs) inside a Confidential Virtual Machine (CVM) configured with AMD SEV-SNP and a standard (non-confidential) virtual machine with similar hardware configuration. The aim is to measure the performance trade-offs introduced by confidential computing while maintaining model functionality and security assurances.

Benchmarking Methodology

The benchmarking was carried out using two identical Google Cloud instances, differing only in the confidentiality configuration:

Non-Confidential VM (baseline) – Standard VM without memory encryption or attestation.

Confidential VM (CVM) – AMD SEV-SNP enabled instance with guest memory encryption and attestation.

Each model was loaded using the Hugging Face transformers library and executed with a fixed prompt length, generating exactly 32 and 128 output tokens per inference. And each inference is run 20 times and their outputs are recorded. The benchmark script recorded:

P50_Latency : Median time it takes for the language model to generate the specified number of tokens under the given batch size and concurrency conditions.

Batch Size : No of tokens processed parallelly.

Token Throughput (Tokens/s): Number of tokens generated per second.

Selected Small Language Models (SLMs)

The evaluation included four representative models of varying sizes and architectures:

distilgpt2 - Compact GPT-2 variant (~82M parameters) [11].

sshleifer/tiny-gpt2 - Extremely small GPT-2 model (~15M parameters) [12].

TinyLlama/TinyLlama-1.1B-Chat-v1.0 - 1.1B parameter model optimised for chat tasks [13].

microsoft/phi-1_5 - 1.3B parameter code-capable model with efficient architecture [14].

Benchmark Results

Table -1 Performance Comparison (Non-Confidential VM vs CVM) at output token set as 32 tokens

Model	Output Tokens	Batch Size	P50_lency		Tokens/s	
			Normal VM	CVM	Normal VM	CVM
distilgpt2	32	8	0.9656	0.9815	265.62	258.04
sshleifer/tiny-gpt2	32	8	0.0505	0.0505	4941.77	5096.17
TinyLlama/TinyLlama-1.1B-Chat-v1.0	32	8	8.6605	9.4223	29.41	27.31
microsoft/phi-1_5	32	8	10.8283	12.0077	23.53	21.28

Table 2 Performance Comparison (Non-Confidential VM vs CVM) at output token set as 128 tokens

Model	Output Tokens	Batch Size	P50_letency		Tokens/s	
			Normal VM	CVM	Normal VM	CVM
distilgpt2	128	8	4.2359	3.8447	242.57	262.42
sshleifer/tiny-gpt2	128	8	0.1964	0.1863	5157.82	5487.98
TinyLlama/TinyLlama-1.1B-Chat-v1.0	128	8	35.6925	37.5071	28.66	27.34
microsoft/phi-1_5	128	8	43.1643	48.7832	23.74	21.12

Analysis of Performance Overheads

The benchmarking results indicate that deploying language models within AMD SEV-SNP Confidential VMs introduces minimal performance overhead. For lightweight models such as *distilgpt2* and *tiny-gpt2*, latency and throughput remained virtually unchanged compared to standard VMs, with some cases even showing slight improvements under CVM. For larger models like *TinyLlama-1.1B* and *phi-1.5*, throughput overheads of ~5-10% and modest latency increases were observed, particularly at longer token lengths. Overall, the findings demonstrate that Confidential Computing environments can secure LLM inference workloads while maintaining near-native performance, making them a practical option for privacy-preserving deployments.

Security vs Performance Trade-Offs

The Confidential VM introduced modest throughput performance overheads for most models (5-10%). In exchange, the CVM ensured **full memory encryption**, **integrity verification**, and **remote attestation**, making it highly suitable for workloads involving sensitive inputs or proprietary models.

Conclusion

The experimental evaluation demonstrates that deploying language models within Confidential Virtual Machines (CVMs) using AMD SEV-SNP introduces only a modest and often negligible performance overhead compared to standard virtual machines. For lightweight models such as *distilgpt2* and *tiny-gpt2*, the observed latency and throughput were nearly identical across both environments, with some runs even showing slightly better performance under CVM. For larger models, including *TinyLlama-1.1B* and *phi-1.5*, CVMs incurred a small throughput reduction in the range of 5–10% and minor increases in latency, particularly at higher output token lengths. Importantly, these trade-offs remain within acceptable limits and do not compromise usability.

Since the current results are based on small language models (SLMs), future work should include extensive benchmarking of larger-scale models to fully evaluate the scalability and usability of CVMs for real-world Large Language Model (LLM) deployments. This will provide a clearer picture of performance-security trade-offs when moving from proof-of-concept experiments to production-ready systems.

References

[1] A. Vaswani *et al.*, “Attention Is All You Need,” Aug. 02, 2023, *arXiv*: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.

[2] “Transformer Architectures - Hugging Face LLM Course.” Accessed: Jul. 02, 2025. [Online]. Available: https://huggingface.co/learn/llm-course/en/chapter1/6

[3] “About – Confidential Computing Consortium.” Accessed: Jul. 02, 2025. [Online]. Available: https://confidentialcomputing.io/about/

[4] “ACE: Confidential Computing for Embedded RISC-V Systems.” Accessed: Jul. 02, 2025. [Online]. Available: https://arxiv.org/html/2505.12995v1

[5] A. Ltd, “Arm Confidential Compute Architecture,” Arm | The Architecture for the Digital World. Accessed: Jul. 02, 2025. [Online]. Available: https://www.arm.com/architecture/security-features/arm-confidential-compute-architecture

[6] R.-V. A.-T. T. Group, “Confidential VM Extension (CoVE) for Confidential Computing on RISC-V platforms”.

[7] D. Lee, D. Kohlbrenner, S. Shinde, K. Asanović, and D. Song, “Keystone: an open framework for architecting trusted execution environments,” in *Proceedings of the Fifteenth European Conference on Computer Systems*, Heraklion Greece: ACM, Apr. 2020, pp. 1–16. doi: 10.1145/3342195.3387532.

[8] D. Kaplan, “AMD SEV-SNP: Strengthening VM Isolation with Integrity Protection and More”.

[9] “distilbert/distilgpt2 · Hugging Face.” Accessed: Jul. 02, 2025. [Online]. Available: https://huggingface.co/distilbert/distilgpt2

[10] *virtee/snpguest*. (Jul. 01, 2025). Rust. VirTEE. Accessed: Jul. 02, 2025. [Online]. Available: https://github.com/virtee/snpguest

[11] “distilbert/distilgpt2 · Hugging Face.” Accessed: Aug. 16, 2025. [Online]. Available: https://huggingface.co/distilbert/distilgpt2

[12] “tiny-gpt2.” Accessed: Aug. 16, 2025. [Online]. Available: https://www.promptlayer.com/models/tiny-gpt2

[13] “TinyLlama/TinyLlama-1.1B-Chat-v1.0 · Hugging Face.” Accessed: Aug. 16, 2025. [Online]. Available: https://huggingface.co/TinyLlama/TinyLlama-1.1B-Chat-v1.0

[14] “microsoft/phi-1_5 · Hugging Face.” Accessed: Aug. 16, 2025. [Online]. Available: https://huggingface.co/microsoft/phi-1_5