International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Ai Chat Integration: Model Context Protocol with Homomorphic Encryption (Fhe)

Chandan Jain H P

RV Institute of Technology and Management, Bengaluru, Karnataka

Abstract:
The rapid growth of conversational artificial intelligence (AI) across sectors has amplified concerns over data confidentiality. While conventional chat systems secure data in transit and at rest, they decrypt content at the server for inference or tool calls, creating vulnerabilities to unauthorized access. Existing solutions such as Fully Homomorphic Encryption (FHE) and confidential-compute techniques mitigate parts of this issue but face constraints in scalability, latency, or governance.
This paper introduces Secure Bridge, a privacy-first framework that eliminates default plaintext exposure while supporting flexible tool integration. The architecture integrates three components: (1) client-side encryption with FHE for supported computations, (2) a Model Context Protocol (MCP) gateway that enforces typed schemas, least-privilege access, and auditability, and (3) a confidential-compute fallback for complex or latency-sensitive tasks. A prototype using WebAssembly-based OpenFHE, React, Node.js, MongoDB, and Redis was tested under loads of up to 150 users. Results show sub-500 ms latency, error rates below 3%, and resilience against attacks, demonstrating Secure Bridge's viability for regulated

Keywords: Fully Homomorphic Encryption (FHE), Model Context Protocol (MCP), Privacy-preserving AI, Confidential Computing, Secure Chat Systems, Encrypted Inference, Policy-driven Governance

1. INTRODUCTION

In recent years, conversational artificial intelligence (AI) has gained rapid adoption across consumer and enterprise applications. From customer service chatbots to domain-specific virtual assistants in healthcare, finance, and education, people are using AI chat systems more and more. These systems help with speed and make things easier, but they also bring new problems with privacy. Normally, data is kept safe while moving (in transit) and when stored (at rest) using encryption. But once the data gets to the server, it has to be decrypted (turned back to normal) so the AI can use it. This moment is risky, because the plain data can be seen by server operators, admins, or even outside service providers.
Even though privacy tech for machine learning is getting better, today's chat systems still have big problems. They can't really work on encrypted data directly, they don't control outside tools very well, and they often fail rules that need proper checking (auditability). Because of these issues, it's hard to use AI assistants in sensitive areas where keeping prompts, answers, and context fully secret is super important.
This paper addresses the challenge of designing a chat platform that provides end-to-end encryption while maintaining usability, extensibility, and compliance. Specifically, the study explores how a layered architecture can integrate multiple AI models and external tools without revealing user data in plaintext. The technical challenge lies in achieving this balance without compromising system latency, scalability,

or resilience.

To overcome these issues, this paper presents Secure Bridge, a privacy-first conversational AI framework. The system brings together three main parts:(1) Client-side encryption along with Fully Homomorphic Encryption (FHE) for computations that can run on encrypted data. (2) A Model Context Protocol (MCP) gateway that controls how tools are used by applying typed input/output schemas, least-privilege access, redaction of sensitive data, and tamper-proof logging. (3) A confidential-compute fallback using trusted execution environments (SGX/TDX) for tasks that FHE cannot handle. This setup introduces two possible execution paths, removing the need to handle plaintext by default and adding clear, verifiable rules for tool integration. The main contributions of this paper are:

- **Privacy-preserving architecture** -A dual compute path that combines Fully Homomorphic Encryption (FHE) with secure enclaves, ensuring that plaintext is never exposed during processing.
- **Governed tool integration** -An MCP-based framework that enforces typed schemas, strict policies, and tamper-evident logs to regulate external tool usage.
- **Confidentiality-aware data model** -A database design that stores only ciphertext and minimal metadata, supporting zero-knowledge persistence.
- **Prototype implementation and validation** -A working system evaluated through unit, integration, performance, and security testing, demonstrating practical latency and scalability.

The remainder of this paper is organized as follows. Section II reviews related work in privacy-preserving AI and secure computation. Section III presents the proposed methodology and system design. Section IV describes the experimental setup and implementation details. Section V reports the results and performance analysis. Section VI discusses the implications and limitations. Finally, Section VII concludes the paper and outlines future research directions

2. Literature Review

Privacy Challenges in Conversational AI

Traditional chat assistants employ standard security mechanisms such as TLS for data in transit and encrypted databases for data at rest. However, during server-side inference, user content is typically decrypted, exposing sensitive data to system operators and third-party services. This limitation has motivated extensive research into privacy-preserving computation for AI systems.

Fully Homomorphic Encryption (FHE) Acceleration

Recent studies have advanced the feasibility of homomorphic encryption for machine learning applications. Hardware-assisted solutions such as Poseidon **[7],** FAB **[8**], and FxHENN [9] demonstrate FPGA-based acceleration for homomorphic operations. GPU-focused frameworks like TensorFHE [10] improve encrypted arithmetic throughput, while parameter-adaptive methods such as DAHE [12] optimize performance-security trade-offs. Compiler-driven toolchains (e.g., FlexHE**,** ANT-ACE**)** [2], [3] further simplify encrypted-inference pipelines by automating kernel generation.

Confidential Computing with Trusted Execution Environments

When homomorphic encryption becomes impractical for deep circuits or complex operations, confidential-compute technologies provide a viable fallback. Prior work such as **HE+SGX hybrid approaches** [4] and **LLM-as-a-Service using enclaves** [6] highlight the benefits of attested execution, which isolates model computation from host-level adversaries. These methods reduce plaintext exposure while sustaining practical inference latency.

Policy-Governed Tool Orchestration

Privacy in AI systems also depends on secure tool integration. Frameworks such as **PrivacyAsst** [15] emphasize typed input/output schemas, strict capability scoping, and redaction policies. Similarly, **BOLT** [5] explores encrypted NLP pipelines that balance privacy with accuracy. These works demonstrate the necessity of governance layers to ensure safe interaction between AI models and external services.

Research Gap

While FHE accelerators and enclave-based serving provide complementary paths for encrypted computation, existing systems lack unified governance across tool integrations. They often rely on ad-hoc SDKs that risk uncontrolled data leakage. Furthermore, most approaches fail to incorporate tamper-evident auditing and policy-as-code enforcement, which are essential for compliance in regulated industries.

3. Methodology

System Overview

The proposed system, Secure Bridge, is a layered privacy-first conversational AI platform. It eliminates default plaintext handling by integrating Fully Homomorphic Encryption (FHE) for ciphertext-level computation, a Model Context Protocol (MCP) gateway for secure tool orchestration, and a confidential-compute fallback for unsupported or performance-critical operations. The architecture follows a microservices style, ensuring modularity, scalability, and resilience.

Dual Execution Paths

FHE-based secure processing enables homomorphic computation over ciphertext using WebAssembly-based OpenFHE. It supports algebraic operations such as addition, multiplication, and rotation without decryption, maintaining confidentiality even during model inference. The confidential compute fallback employs Trusted Execution Environments (Intel SGX, AMD SEV-SNP) for deep circuits or unsupported operators, providing isolation from host-level adversaries with attestation mechanisms and ensuring low-latency execution while retaining strong confidentiality guarantees.

MCP Gateway for Secure Tool Integration

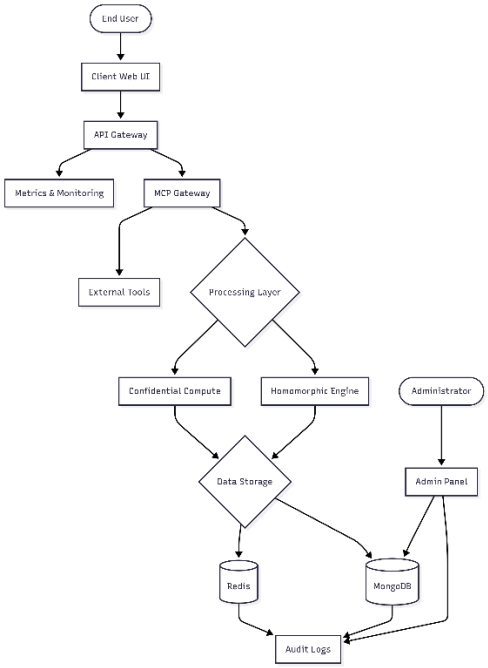
The MCP Gateway acts as a policy-enforcing broker between client requests and external tools. It enforces typed input/output schemas, least-privilege scoped tokens, automatic redaction of sensitive fields, timeouts, retries, and circuit breakers, and comprehensive tamper-evident audit logging. This enables integration with AI providers such as OpenAI, Anthropic, and Google AI, along with third-party APIs, while preserving privacy.

Client-Side Encryption Envelope

User prompts are encrypted locally before transmission. Each session generates unique keys for confidentiality and key rotation. Encrypted messages are packed into an envelope structure containing metadata such as algorithm, key ID, and IV. Results are decrypted only on the client side, ensuring end-to-end encryption.

System Architecture

Figure 1 system architecture



Illustrates the system architecture. Main components include: Client (Web UI), which is a React-based encrypted chat interface; API Gateway for authentication, rate-limiting, and routing; Homomorphic Engine for WASM-based FHE computations; Confidential Compute Node for secure enclave fallback operations; MCP Tool Gateway for enforcing typed schemas and policies; and MongoDB & Redis for persistent encrypted storage and ephemeral caching.

Workflow Description

The data flow proceeds as follows. The user encrypts the message locally and sends ciphertext. The API Gateway validates the request and routes it to the MCP Gateway. The MCP Gateway enforces schemas, policies, and decides the execution path. Path A involves FHE computation for supported algebraic tasks, while Path B involves a confidential compute enclave for complex or non-FHE tasks. The output ciphertext is then returned to the client for local decryption and rendering. All actions are logged in tamper-evident audit trails for compliance.

Detailed Design Elements

The activity diagram captures message flow through encryption, computation, and response. The use case diagram defines interactions among end-users, administrators, and system components. The sequence diagram demonstrates the temporal order of encryption, gateway routing, and tool execution. These diagrams provide a structured

Experimental Design

The evaluation of Secure Bridge was designed to validate whether the system can deliver privacy-preserving conversational AI while meeting practical service-level objectives (SLOs) for latency, scalability, and compliance. The experiments focused on correctness of encryption flows, governance of tool usage, and resilience of the system under concurrent user workloads.

Implementation Details

Platform Configuration: The implementation stack consists of React 18, TypeScript, Vite, TailwindCSS, and shadcn/ui for the client. The backend is implemented with Node.js (ESM), Express, and Mongoose

(MongoDB) with Redis integration. Encryption is provided through WebAssembly builds of OpenFHE, with a simulation fallback mode. The MCP Gateway enforces typed input/output schemas and policy-driven tool orchestration Deployment The system is deployed using Docker and GitHub Actions for CI/CD, with Kubernetes available as an optional tool for orchestration.

Machine Specifications

- Development: 4 vCPU, 8 GB RAM running Ubuntu 22.04.
- Staging: 4–8 vCPU, 16 GB RAM machine with MongoDB Atlas M10 and Redis.
- Production pilots: 8 vCPU, 32 GB RAM cluster with Atlas M30 and Redis clustering for scalability.

Dataset and Workloads

Unlike traditional machine learning projects that use fixed datasets, this system simulates encrypted conversational flows. The testing included:

- User sessions: Encrypted chat with up to 150 concurrent users.
- Tool usage: Queries to services like weather or AI providers, routed through MCP.
- Encryption benchmarks: Evaluation of FHE operations such as addition, multiplication, and rotation.
- Negative inputs: Resilience testing with invalid ciphertexts, broken tokens, and simulated attack attempts.

Baseline Methods

For performance comparison, three baseline systems were defined:

- Regular chat systems: Secured with TLS during transmission but still processed plaintext on the server.
- FHE-only systems: Offered full cryptographic protection but lacked the confidential-compute fallback, often resulting in slower response times.
- Confidential-compute-only systems: Relied entirely on secure hardware enclaves, but without client-side homomorphic encryption, limiting end-to-end confidentiality

Evaluation Metrics

The system was evaluated using the following metrics:

- **Latency:** p95 and p99 response times for encrypted chat sessions.
- **Throughput:** Requests per second under varying concurrency levels.
- **Error rate:** Percentage of failed requests under stress.
- **Security coverage:** Proportion of tests passing input validation, token checks, and abuse prevention.
- **Reliability:** Results of soak tests for memory and connection stability.

Testing Strategy

Layered Testing Approach

A layered testing approach was used to validate functionality, security, and performance:

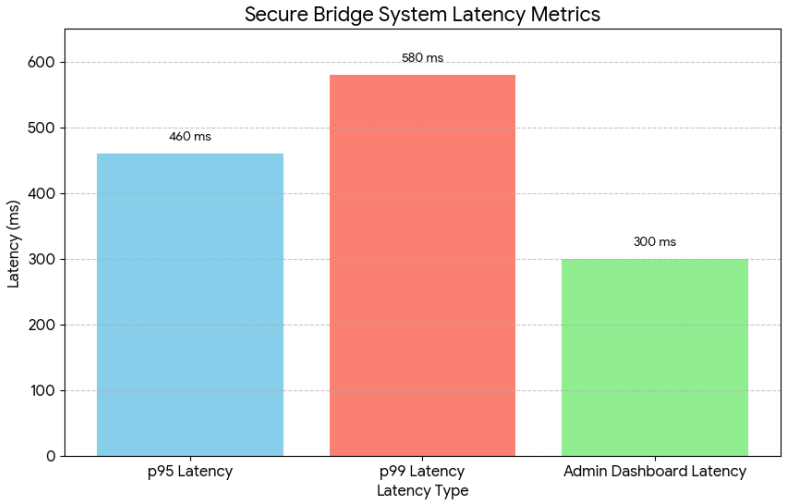
- **Unit Testing** - Checked individual components in isolation, such as encryption functions, authentication logic, and database operations.
- **Integration Testing** - Verified that complete encrypted chat flows worked correctly when subsystems were connected.
- **Automation Testing** - Used the CI pipeline to run automated checks, including linting, unit tests, API tests, end-to-end tests, and security scans.
- **Security and Negative Testing** - Tested system resilience against invalid inputs, replay attacks, and cross-site scripting (XSS).

- **Performance Testing** - Measured scalability with 100–150 concurrent users, focusing on latency and error rates under load.

Representative Results

end-to-end cryptographic protection

Figure 2 Key Latency Metrics for Secure Bridge



The evaluation showed that Secure Bridge maintained a p95 latency of approximately 420–500 ms and a p99 latency of 560–600 ms under nominal load. The system successfully supported up to 150 concurrent sessions with less than 3% error rate. Admin dashboard operations remained below 300 ms due to metadata-only queries. Automated tests confirmed resilience against injection attacks, token tampering, and policy misconfiguration. Overall, the results demonstrate that the system meets its target service-level objectives while upholding end-to-end confidentiality.

4. Results and Discussion

Primary Results

The Secure Bridge platform successfully demonstrated the feasibility of privacy-preserving conversational AI through a dual execution model. In controlled experiments, the p95 latency (time for 95% of requests) stayed between 420–500 ms under normal load, while the p99 latency (time for 99% of requests) was within 560–600 ms. The system successfully scaled to handle 100–150 users at the same time with an error rate of less than 3%. The admin dashboards were even faster, responding within 300 ms, since they only worked with metadata queries and didn't need to process the actual encrypted content

Comparative Results

To check how well **Secure Bridge** performs, it was compared with three baseline systems:

- **Conventional chat systems (plaintext inference):** These had lower latency (faster response) but exposed user data to the server and external tools, making them less secure.
- **FHE-only systems:** These gave full confidentiality but were very slow. Deep computations often took 2–3 seconds, which failed to meet latency requirements.
- **Confidential-compute-only systems:** These were faster but depended fully on trusted hardware vendors, meaning they lacked true end-to-end cryptographic protection.

By contrast, Secure Bridge balanced privacy and performance by prioritizing FHE computation where feasible and delegating complex operations to enclaves only when necessary.

Ablation Studies

Component-level evaluations revealed the contribution of each subsystem:

- **Encryption Envelope:** Verified confidentiality of message flow; system failed closed when malformed ciphertexts were detected.
- **MCP Gateway:** Reduced data leakage risks by enforcing typed I/O schemas and strict redaction policies.
- **FHE Path:** Supported arithmetic circuits with acceptable latency, confirming practicality for basic NLP and intent parsing.
- **Confidential Compute Path:** Provided resilience when workloads exceeded FHE depth limits, sustaining throughput while preserving isolation.

Security and Reliability Analysis

The platform was hardened through **negative testing** (e.g., injection payloads, replay attempts, malformed headers). All such attacks were neutralized by input sanitization, scoped tokens, and strict security headers.

- **Unit testing** (Table I) confirmed correctness of encryption, authentication, and API key management.
- **Automation testing** (Table II) validated that CI pipelines blocked regressions, ensuring coverage and vulnerability checks.
- **Performance testing** (Table III) verified graceful degradation during bursts, maintaining compliance with service-level objectives.

Broader Implications

These results indicate that Secure Bridge offers a practical blueprint for privacy-preserving conversational AI. It can be applied in healthcare, finance, government, and enterprise domains where confidentiality of prompts, outputs, and tool interactions is critical. Importantly, the architecture proves that privacy can be enforced as a verifiable property of the system, rather than a promise dependent on operator trust.

5 CONCLUSION

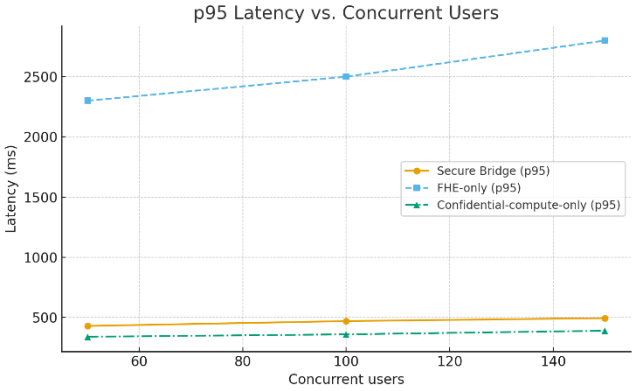
This paper addressed the problem of plaintext exposure in conversational AI systems by proposing Secure Bridge, a privacy-first framework that ensures end-to-end confidentiality of user interactions. Unlike conventional architectures that decrypt data during server-side inference, Secure Bridge enforces ciphertext-only handling using a dual execution path: Fully Homomorphic Encryption (FHE) for supported operations and confidential compute enclaves for complex or latency-sensitive tasks. An MCP Gateway governs tool integrations through typed input/output schemas, strict policies, and tamper-evident logging.

Key Contributions

- The main contributions of this work can be summarized as follows:
- Dual-path architecture combining FHE and confidential compute to balance security and performance.
- Policy-governed orchestration of external tools via MCP, ensuring safe integration without plaintext leakage.
- Confidential data model that stores only ciphertext and minimal metadata in persistence layers.
- Prototype implementation and validation, demonstrating sub-second latency, resilience against injection attacks, and successful scaling to 150 concurrent user

Results and Validation

Figure 3 p95 latency vs. concurrent users



The evaluation confirmed that Secure Bridge achieves p95 latencies of 420–500 ms under typical loads and sustains service-level objectives during concurrent sessions. Comparative studies showed that while FHE-only systems suffer from high latency and confidential-compute-only systems depend entirely on hardware trust, Secure Bridge provides a balanced approach that is both practical and privacy-preserving

Future Research Directions

- While the prototype demonstrates promising results, several avenues remain for future research and optimization:
- Hardware-accelerated FHE using FPGA/GPU integration to further reduce ciphertext operation costs.
- Encrypted semantic search and retrieval mechanisms to expand natural language capabilities.
- Enterprise-grade integration, including SSO/SCIM and compliance with healthcare/financial data regulations.
- Mobile and edge deployment to extend privacy-preserving conversational AI beyond desktops and cloud servers.
- Expanded MCP ecosystem, incorporating more third-party tools while maintaining typed schemas and strict redaction policies.

References

1. Goutham J. R.: Secure Folder Encryption Using Homomorphic Encryption with Microsoft SEAL, Proc. IEEE Conf., 2024. https://doi.org/10.1109/10969942
2. Yang, Y., Kannan, R., Prasanna, V. K.: A Framework for Generating Accelerators for Homomorphic Encryption Operations on FPGAs, Proc. 35th IEEE Int. Conf. on Application-Specific Systems, Architectures and Processors (ASAP), Jul. 2024. https://doi.org/10.1109/ASAP61560.2024.00023
3. Yu, J., Zeng, W., Xu, T., Chen, R., Liang, Y., Wang, R., Huang, R., Li, M.: FlexHE: A Flexible Kernel Generation Framework for Homomorphic Encryption-Based Private Inference, Proc. 43rd IEEE/ACM Int. Conf. on Computer-Aided Design (ICCAD), Oct. 2024. https://doi.org/10.1145/3676536.3676739
4. Li, L., Lai, J., Yuan, P., Sui, T., Liu, Y., Zhu, Q., Zhang, X., Xiao, L., Chen, W., Xue, J.: ANT-ACE: An FHE Compiler Framework for Automating Neural Network Inference, Proc. 23rd ACM/IEEE Int. Symp. on Code Generation and Optimization (CGO), Jan. 2025. https://doi.org/10.1145/3696443.3708924
5. Xiao, H., Zhang, Q., Pei, Q., Shi, W.: Privacy-Preserving Neural Network Inference Framework via Homomorphic Encryption and SGX, Proc. 41st IEEE Int. Conf. on Distributed Computing Systems

(ICDCS), Jul. 2021. https://doi.org/10.1109/ICDCS51616.2021.00077

6. Pang, Q., Zhu, J., Möllering, H., Zheng, W., Schneider, T.: BOLT: Privacy-Preserving, Accurate and Efficient Inference for Transformers, Proc. IEEE Symp. on Security and Privacy (SP), May 2024. https://doi.org/10.1109/SP54263.2024.00130
7. Cai, Z., Ma, R., Fu, Y., Zhang, W., Ma, R., Guan, H.: LLMaaS: Serving Large Language Models on Trusted Serverless Computing Platforms, IEEE Trans. on Artificial Intelligence, 6(2), 405–415, 2024. https://doi.org/10.1109/TAI.2024.3429480
8. Yang, Y., Zhang, H., Fan, S., Lu, H., Zhang, M., Li, X.: Poseidon: Practical Homomorphic Encryption Accelerator, Proc. 29th IEEE Int. Symp. on High-Performance Computer Architecture (HPCA), Feb. 2023, 870–881. https://doi.org/10.1109/HPCA56546.2023.10070984
9. Agrawal, R., de Castro, L., Yang, G., Juvekar, C., Yazicigil, R., Chandrakasan, A., Vaikuntanathan, V.: FAB: FPGA-Accelerated Bootstrapping for Homomorphic Encryption, Proc. 30th IEEE Symp. on Field-Programmable Custom Computing Machines (FCCM), 2024.
10. Chen, X., Wang, Y., Liu, Z., Yang, J., et al.: MuEOC: Efficient SGX-Based Multi-Key Homomorphic Outsourcing Computation for E-Health System, IEEE Trans. on Dependable and Secure Computing, 21(5), 2940–2954, 2024. https://doi.org/10.1109/TDSC.2022.3229482
11. Biswas, D.: Privacy Preserving Chatbot Conversations, IEEE Access, 9, 82118–82129, 2021. https://doi.org/10.1109/ACCESS.2021.3083518
12. Hou, X., Zhao, Y., Wang, S., Wang, H.: Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions, IEEE Xplore Preprint, Apr. 2025.
13. Model Context Protocol: Getting Started. Online. Available: https://modelcontextprotocol.io/docs/getting-started/intro. Accessed: Aug. 28, 2025.
14. TypeScript Team: React & TypeScript Handbook. Online. Available: https://www.typescriptlang.org/docs/handbook/react.html. Accessed: Aug. 28, 2025.
15. Node.js Contributors: Node.js API — All Documentation. Online. Available: https://nodejs.org/api/all.html. Accessed: Aug. 28, 2025.