Engineering Science & Technology
https://ojs.wiserpub.com/index.php/EST/

Review Article

Engineering Secure AI Systems with Lattice-Based Cryptography and Large Language Models

Devharsh Trivedi

Department of Computer Science, Bowie State University, Bowie, MD, USA
E-mail: dtrivedi@bowiestate.edu

Received: 26 June 2026; **Revised:** 3 August 2026; **Accepted:** 6 August 2026

Abstract: Deploying large language models in safety-critical domains while migrating to post-quantum cryptographic standards poses a compounded systems engineering challenge: infrastructure must become privacy-preserving, computationally viable, and quantum-resistant at once. This review examines three frontiers where lattice-based cryptography and large language models converge, applying an evidence-grading scheme that separates established results from conjecture. The first frontier is fully homomorphic encryption for private model training. Ciphertext error does not provide differential privacy: it is not calibrated to query sensitivity, involves no gradient clipping, and is removed by decryption. The two mechanisms defend against disjoint adversaries and must be composed, not substituted. The second frontier is the hypothesis that language models can act as adaptive agents inside lattice reduction. No empirical result supports it. The review further shows that scheduling improvements cannot reduce core security estimates, since these depend on the minimum viable block size rather than the number of reduction tours, and supplies a falsifiable benchmark protocol in place of the parameter-selection guidance of earlier treatments. The third frontier is functional encryption for decentralized inference. Inner-product constructions admit exact reconstruction of hidden states by any node holding a spanning key set, requiring no cryptanalysis, so security rests on key allocation rather than lattice hardness. Attention is bilinear and lies outside the inner-product functionality class. Quantitatively, encrypted inference for a small encoder costs roughly five orders of magnitude more than plaintext, measured on a graphics processing unit; functional key material reaches four tebibytes per attention layer under the parameters the security proof requires, reducible by three orders of magnitude at a stated security cost; and memory capacity, not arithmetic throughput, is the binding constraint. Readiness levels for the three frontiers span one to three. Trusted execution environments remain the only approach in production today.

***Keywords*:** lattice-based cryptography, large language models, fully homomorphic encryption, functional encryption, post-quantum cryptography, computational engineering, privacy-preserving AI

Abbreviation

Acronyms

ABE Attribute-Based Encryption
ALS Agrawal-Libert-Stehlé IPFE Scheme

DOI: https://doi.org/10.37256/est.81202710357

ASIC	Application-Specific Integrated Circuit
BFV	Brakerski-Fan-Vercauteren FHE Scheme
BGV	Brakerski-Gentry-Vaikuntanathan FHE Scheme
BKZ	Block Korkine-Zolotarev Lattice Reduction
CKKS	Cheon-Kim-Kim-Song Approximate FHE Scheme
CVP	Closest Vector Problem
DMCFE	Decentralized Mcfe
DP	Differential Privacy
FE	Functional Encryption
FHE	Fully Homomorphic Encryption
FL	Federated Learning
FPGA	Field-Programmable Gate Array
GSA	Geometric Series Assumption
GSO	Gram-Schmidt Orthogonalization
HNDL	Harvest Now, Decrypt Later
IBE	Identity-Based Encryption
IPFE	Inner-Product Functional Encryption
LBC	Lattice-Based Cryptography
LLL	Lenstra-Lenstra-Lovász Reduction
LLM	Large Language Model
LORA	Low-Rank Adaptation
LWE	Learning with Errors
MCFE	Multi-Client Functional Encryption
MIA	Membership Inference Attack
ML-DSA	Module-Lattice Digital Signature Algorithm (FIPS 204)
ML-KEM	Module-Lattice Key-Encapsulation Mechanism (FIPS 203)
MLWE	Module Learning with Errors
MPC	Secure Multi-Party Computation
NTT	Number-Theoretic Transform
PET	Privacy-Enhancing Technology
PQC	Post-Quantum Cryptography
RLWE	Ring Learning with Errors
SIS	Short Integer Solution Problem
SLH-DSA	Stateless Hash-Based DSA (FIPS 205)
SVP	Shortest Vector Problem
TEE	Trusted Execution Environment
TFHE	Fast Fully Homomorphic Encryption Over the Torus
TRL	Technology Readiness Level
IND-CPA	Indistinguishability Under Chosen-Plaintext Attack
SL	Split Learning

Notation

n	Lattice dimension, or LWE secret dimension
q	Ciphertext modulus
σ	Standard deviation of the error distribution
χ_σ	Discrete Gaussian error distribution with parameter σ
Λ	A lattice
$\boldsymbol{B}$	Lattice basis; $\boldsymbol{b}_i$ its i-th vector

$\widetilde{\boldsymbol{b}}_i$	i-th Gram-Schmidt vector (written $\boldsymbol{b}_i^*$ in Section 5, following BKZ convention)
β	BKZ block size; β^* the minimum viable block size for an attack
δ	Root Hermite factor (distinct from the DP parameter δ)
$\varepsilon,\ \delta$	Differential privacy parameters
Δf	ℓ_2-sensitivity of a query f
R_q	Polynomial ring $\mathbb{Z}_q[X]/\left(X^N+1\right)$
N	Ring degree in FHE; k the module rank in ML-KEM
Δ	CKKS scaling factor
d_{model}	Transformer hidden dimension; d_k per-head dimension
L	Sequence length, or network depth where indicated
r	LoRA rank
$\boldsymbol{h}$	Hidden state (activation) vector
sk_y	Functional decryption key for the vector $\boldsymbol{y}$
λ	Security parameter, in bits

Two symbols are unavoidably overloaded by convention and are disambiguated by context throughout: σ denotes both the root Hermite factor in lattice reduction and the failure probability in differential privacy, and Δ denotes both the CKKS scaling factor and the ℓ_2-sensitivity in the DP literature. Where confusion is possible, the intended reading is stated explicitly

1. Introduction

The modern digital infrastructure sits at the confluence of two significant shifts. Large Language Models (LLMs) such as the GPT family, LLaMA, and Gemini have demonstrated remarkable capabilities across natural language understanding, code generation, and scientific reasoning, yet they raise serious concerns around data privacy, intellectual property, and model security [1]-[3]. Concurrently, quantum computing advances have placed the long-term viability of classical public-key cryptography under existential threat, accelerating the global standardization of post-quantum cryptographic primitives [4], [5].

Lattice-Based Cryptography (LBC) has emerged as the dominant post-quantum paradigm. Its security rests on hard computational problems in high-dimensional integer lattices, most notably the Learning with Errors (LWE) problem and its Ring Learning with Errors (RLWE), both of which are believed to resist quantum attacks [6], [7]. Beyond key encapsulation and digital signatures, LBC enables Fully Homomorphic Encryption (FHE) [8]-[10] and Functional Encryption (FE) [11], which allow computation on encrypted data with strong, lattice-hardness-based security proofs.

The intersection of LBC and LLMs addresses fundamental engineering problems. Training LLMs on sensitive corpora, including medical records, legal documents, or proprietary code, exposes data to memorization and leakage risks. Deploying LLMs at the inference layer in untrusted or decentralized environments requires strong guarantees that neither the model weights nor the user prompts are exposed. Conversely, the heuristic and pattern-recognition capabilities of LLMs offer a novel lens through which to approach hard lattice problems that underpin the security assumptions of post-quantum cryptography itself. Empirical demonstrations of this intersection are now available in the literature. Polynomial approximation techniques for encrypted inference have been shown sufficient for practical security analytics workloads under CKKS [12], and privacy-preserving mechanisms for distributed ML pipelines have been demonstrated in federated split-learning settings [13]. Broader studies on security analytics and on the practical trajectory of post-quantum migration situate these results in deployed settings.

The engineering disciplines most directly implicated are computational engineering, which must absorb the factor-of-one-thousand-to-ten-thousand throughput penalty of homomorphic operations into system budgets; informatics and communication engineering, which must propagate post-quantum key encapsulation and digital signature standards through deployed protocol stacks; and electrical and electronics engineering, which must build the custom application-specific integrated circuits and field-programmable gate array accelerators that make LWE arithmetic fast enough for real-time inference. This review is structured to be useful to practitioners in all three communities: it provides the mathematical foundations required to evaluate security claims, the implementation complexity data required to budget

engineering resources, and Technology Readiness Level (TRL) assessments that map each construction to a deployment horizon.

1.1 *What distinguishes this review*

Several strong surveys already cover parts of this territory, and it is worth stating precisely where this one differs. Peikert's decade survey [5] and the Micciancio-Goldwasser monograph [14] primarily focus on mathematical foundations rather than engineering deployment or ML applications. The concrete hardness literature [15], [16] is authoritative on parameter selection but does not consider privacy-preserving inference. Existing surveys of privacy-preserving machine learning and of practical post-quantum migration each address one side of the intersection. What is missing, and what this review supplies, is a treatment that holds the two together: a single account in which one family of lattice hardness assumptions is traced from the standardized key-encapsulation mechanism, through the homomorphic encryption scheme, to the functional encryption construction, with engineering budgets and readiness assessments attached at each step.

Table 1 makes the comparison explicit. The distinguishing contributions of this review are fourfold: the algebraic bridge of Section 3.3, which shows non-cryptographers why one hardness assumption yields three apparently unrelated capabilities; the comparative evaluation of Section 7, which positions lattice methods against the trusted-execution, multi-party, and federated alternatives that compete for the same engineering budget; the explicit evidence grading of Section 2.4, which separates what is known from what is conjectured; and the prioritized roadmap of Section 10.

Table 1. Coverage of this review relative to prior surveys. Full coverage (•) indicates the topic is treated as a primary subject; partial (○) indicates it appears as context; absent (−) indicates it is out of scope for that work

Topic	[5]	[14]	[15]	[17]	This
Lattice hardness foundations	•	•	•	○	•
FHE constructions	○	–	–	○	•
Functional encryption	○	–	–	–	•
Concrete parameter selection	○	–	•	–	•
Transformer/LLM workloads	–	–	–	–	•
LLM-assisted cryptanalysis	–	–	–	–	•
Comparison with MPC/TEE/FL	–	–	–	–	•
Hardware co-design	–	–	–	•	•
Readiness assessment	–	–	–	–	•
Explicit evidence grading	–	–	–	–	•

This review identifies and structures three primary engineering frontiers at the LBC-LLM intersection, as illustrated in Figure 1. Each frontier arises from a different role that lattice mathematics can play in relation to LLMs: as a privacy guarantor, as a target for intelligent attack, and as an access-control mechanism. Together they span the full threat model facing AI systems in a post-quantum world. The August 2024 National Institute of Standards and Technology (NIST) finalization of FIPS 203, FIPS 204, and FIPS 205 [4], [18], [19] makes the practical relevance of this intersection acute. ML-KEM and ML-DSA rest on Module-LWE, CKKS on Ring-LWE, and the Agrawal-Libert-Stehlé (ALS) inner-product scheme on plain LWE. These are three points on one spectrum rather than one assumption, and the distinction matters both for security analysis and for engineering; it is made precise in Section 3.3. What they share is a single family of hard problems, which means the standardization of the first two carries direct implications for the deployability of the others.

The first frontier, examined in Section 4, analyzes FHE built on LWE as a mechanism for privacy-preserving LLM training. The second frontier, examined in Section 5, surveys the paradigm of using LLMs as adaptive heuristic agents inside lattice reduction algorithms, with implications for post-quantum security estimation. The third frontier, Section 6, analyzes lattice-based FE for fine-grained, verifiable computation over transformer layers in decentralized inference pipelines. Section 7 positions all three against the competing privacy-enhancing technologies with which a systems

engineer must actually choose, Section 8 presents a comparative taxonomy, Section 9 treats deployment realities, Section 10 prioritizes open problems into an engineering roadmap, and Section 11 concludes.

Section 2 states the review methodology, including the search strategy and the evidence-grading scheme used to separate established results from the hypotheses this review advances. Readers are urged to consult the evidence register in Table 2 before interpreting any quantitative claim below.

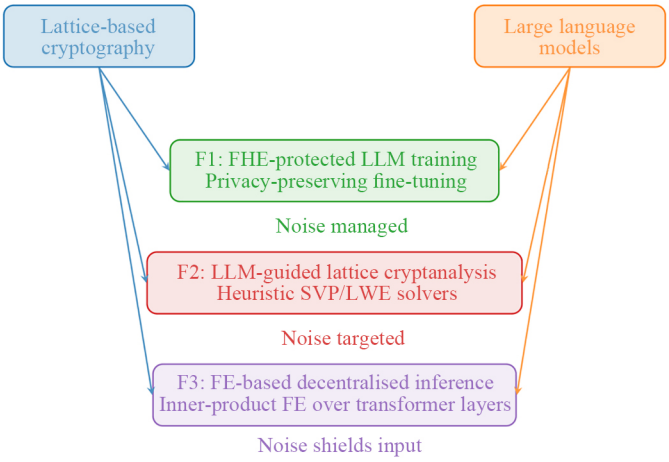

Figure 1. The three engineering frontiers surveyed in this paper, organized around the central tension between LBC security properties (hardness, noise, and functional correctness) and LLM capabilities (pattern recognition, fine-tuning, and distributed inference)

Table 2. Evidence register. Each substantive claim in this review is assigned an evidence class. Frontier 2 is, in its entirety, a research proposal (E4): no LLM-guided lattice reduction experiment is reported here or, to the authors' knowledge, anywhere in the published literature

Section	Claim	Class
3.1	LWE and RLWE hardness; worst-case to average-case reduction; LLL and BKZ behaviour	E1
3.2	ML-KEM, ML-DSA, SLH-DSA parameter sets and standardization status	E1
4.2	CKKS, BFV, BGV, TFHE construction and noise growth behaviour	E1
4.5	Homomorphic throughput penalty for transformer primitives	E2
4.4	Ciphertext error does *not* provide differential privacy	E1
4.3	Encrypted LoRA fine-tuning is realisable at production scale	E4
5.2	BKZ cost model and core-SVP security estimation	E1
5.3	LLMs can act as adaptive block-size schedulers for BKZ	E4
5.4	Any specific quantitative margin for LLM-assisted attacks	E4
6.2	ALS inner-product FE correctness and IND-CPA security	E1
6.2	Correctness degradation of a toy IPFE instance under increasing noise	E3
6.4	Inner-product FE alone suffices for transformer attention	Refited (Sec. 6.3.1)
7	Relative standing of FHE, MPC, TEE, FL, DP	E2
8.4	Technology Readiness Level assignments	E4 (author-assigned; criteria in Table 17)

2. Review methodology

A recurring weakness of surveys at the boundary between cryptography and machine learning is that established results, single-source benchmarks, and authorial conjecture are presented in a uniform declarative voice, leaving the reader unable to judge which claims carry evidentiary weight. This section states the search strategy, eligibility criteria,

and evidence-grading scheme used here, and closes with a claim-level register (Table 2) that assigns every substantive assertion in this review to an explicit evidence class.

2.1 *Search strategy*

Literature was retrieved from six sources: the International Association for Cryptologic Research (IACR) Cryptology ePrint Archive, IEEE Xplore, the Association for Computing Machinery (ACM) Digital Library, arXiv (cs.CR, cs.LG, cs.CL), DBLP, and Google Scholar for forward-citation tracing. The search window runs from 2005, the year of Regev's introduction of LWE [6], through the first quarter of 2026. Standards documents and agency guidance were retrieved directly from the issuing bodies rather than through bibliographic databases.

Queries were constructed as the Cartesian product of a cryptographic term set and an application term set. The cryptographic set comprised: learning with errors, ring-LWE, module-LWE, lattice reduction, BKZ, fully homomorphic encryption, CKKS, BFV, BGV, TFHE, functional encryption, inner-product encryption, and post-quantum migration. The application set comprised: transformer, large language model, private inference, encrypted training, LoRA, fine-tuning, split learning, federated learning, hardware acceleration, NTT, and Graphics Processing Unit (GPU). Backward and forward citation chasing was applied to every included primary construction.

2.2 *Eligibility criteria*

Sources were included if they satisfied all of the following: (i) they present a cryptographic construction, a security analysis, a measured benchmark, or a standardization decision; (ii) they concern lattice-based primitives or the privacy-preserving evaluation of neural network workloads; and (iii) they report sufficient parameter detail for a reader to situate the result within a concrete security level. Preprints were admitted, since a substantial fraction of the relevant cryptanalytic literature appears only on ePrint, but are marked as such in the evidence register and are never used as the sole support for a security-relevant claim.

Sources were excluded if they: report performance figures without stating ring dimension, ciphertext modulus, or claimed security level; concern non-lattice post-quantum families (code-based, isogeny-based, multivariate) except where required for comparative context; or present privacy-preserving machine learning without a cryptographic hardness assumption, such as heuristic obfuscation or anonymization-only pipelines.

2.3 *Scope boundaries and exclusions*

This review deliberately does not cover several adjacent areas, and readers should not interpret its silence as a judgement of their importance. Excluded from scope are: hash-based and code-based post-quantum signatures beyond the comparative note in Section 3.2; the full formal-verification literature for cryptographic implementations; side-channel and fault-injection attacks against lattice implementations, which constitute a substantial literature in their own right; adversarial robustness, prompt injection, and alignment failures in LLMs, which are orthogonal to the confidentiality properties considered here; regulatory and legal compliance frameworks; and quantum algorithms for lattice problems beyond the extent needed to justify the post-quantum security premise.

2.4 *Evidence grading*

Every substantive claim in this review is assigned one of four evidence classes:

• Established. A peer-reviewed result that has been independently reproduced, standardized, or is treated as settled by the cryptographic community. Example: the security reduction from LWE to worst-case lattice problems.

• Reported. A peer-reviewed or archived benchmark from a single research group, not independently replicated. Figures in this class should be read as order-of-magnitude indicators that depend on the originating hardware and parameter selection.

• Original simulation. A small-scale computation performed for this review and released with the accompanying notebook. These use toy parameters that are far below cryptographic scale and are included to illustrate qualitative behaviour, not to establish performance claims.

• Hypothesis. A conjecture advanced by this review for which no supporting experiment exists, here or elsewhere. Claims in this class are stated as open research questions and are accompanied by a falsifiable evaluation protocol wherever possible.

The register below is the interpretive key for the remainder of the paper. A reader who wishes to know what this review actually demonstrates, as opposed to what it organizes or proposes, should read only the E1 and E3 rows.

2.5 *Reproducibility of the original simulations*

Every figure carrying evidence class E3 is produced by the accompanying notebook, which is distributed with this manuscript and contains the complete source for each construction, sweep, and plot. The following details are recorded so that any reported number can be checked or contradicted.

2.5.1 *Environment*

Python 3.14.6 with NumPy 2.4.6 and Matplotlib 3.10.9, on macOS 15.7.8 (x86-64), executed natively on an Intel Core i5-8279U at 2.40 GHz with no GPU. All timings are single-threaded wall-clock measurements taken with time. perf_counter after warm-up iterations. Absolute timings should be expected to differ on other hardware; only the ratios reported in Figure 2 are of interest, and those carry the run-to-run variance stated in that caption.

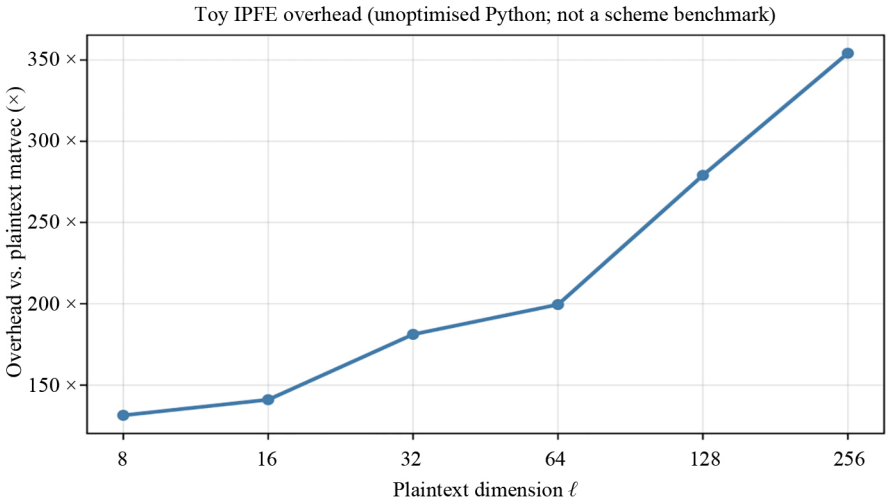

Figure 2. Measured overhead of the toy IPFE linear projection relative to plaintext matrix-vector multiplication as the plaintext dimension ℓ grows, for $d_{\text{out}} = 8$ output rows, 30 repetitions after warm-up. The observed range is of order $10^2\times$, specifically $10^3\times$ to $246\times$ in the run plotted here, which is the same run stored in the accompanying notebook. Across repeated runs and across machines the range has varied roughly $80\times$ and $360\times$. Because the plaintext baseline is a NumPy matrix-vector product of a few microseconds, run-to-run timing variance is a substantial fraction of the measurement, and the ratio should be read only to within a factor of about two. An earlier version of this review stated 10^3-$10^4\times$ in the caption, which contradicted the plotted data; the figure now reports what was measured. Two cautions apply. First, this is an unoptimised Python reference implementation, so the ratio reflects interpreter overhead as much as scheme cost and is not a benchmark of the ALS construction. Second, the plaintext baseline is a NumPy matvec of a few microseconds, so small absolute differences produce large ratios. The figure supports only the qualitative claim that overhead grows with ℓ because each output coordinate requires an independent decryption. Evidence class E3

2.5.2 *Randomness*

All stochastic components draw from a seeded NumPy Generator; seeds are fixed in the notebook and are not resampled between runs, so every figure whose content is determined by the random draw regenerates identically. Figure

2 is the exception: it reports wall-clock timings, which depend on the machine and on scheduling rather than on the seed, and it therefore does not reproduce exactly. Its caption states the range observed across runs.

2.5.3 *Parameters*

The toy IPFE instance uses $n = 10$, $m = 40$, $q = 131,072$, bound $B = 256$, and plaintext dimension ℓ as stated per figure, with $\boldsymbol{x}$ and $\boldsymbol{y}$ entries uniform on $\{2, \ldots, 2\}$. The lattice reduction experiments use dimension 30 with a random q-ary basis. These parameters are three orders of magnitude below cryptographic scale and were chosen so that exhaustive sweeps complete in minutes. What these simulations do not establish.

They are reference implementations in interpreted Python, not optimized libraries. Timing ratios derived from them reflect interpreter overhead as much as algorithmic cost and must not be read as benchmarks of the underlying schemes; the published figures in Section 4.5, which carry class E2, are the appropriate source for performance claims. No result in this review depends on the absolute timings reported from the notebook.

2.5.4 *Estimator settings*

Security estimates quoted for standardized parameter sets are taken from the published literature and from the LWE estimator [15] under the core-SVP model with sieving coefficient 0.292. The simplified closed-form estimator implemented in the notebook is calibrated against published values and is included for illustration of trends only; where the two disagree, the published estimates govern.

2.6 *Distinguishing contribution types*

Three kinds of material appear in this review and are typographically and structurally separated throughout. Synthesis reorganizes published results and is the dominant mode in Sections 3, 7, and 8. Original simulation appears only in the figures explicitly marked E3 in Table 2, all of which derive from the accompanying notebook at toy parameters. Proposal covers the encrypted fine-tuning protocol of Section 4.3, the LLM-guided reduction paradigm of Section 5, and the decentralized inference architecture of Section 6.4; none of these has been implemented at cryptographic scale, and each is accompanied by a statement of the specific circuits, constructions, or experiments that would be required to move it from proposal to result.

3. Background

The convergence of lattice-based cryptography and large language models is driven by concrete, practical gaps in the security architecture of every database and AI deployment in production today. Traditional database security is organized around the Confidentiality, Integrity, and Availability (CIA) triad of confidentiality, integrity, and availability, with a fourth accountability pillar added in modern frameworks. Encryption at rest (AES-256) and encryption in transit (TLS 1.3) are now standard practice, but a critical architectural gap persists: data must be decrypted into plaintext in memory before any computation, query, or inference step can execute. This vulnerability, sometimes called the data-in-use gap, is precisely what fully homomorphic encryption is designed to close [8], making FHE the cryptographic foundation for the first frontier surveyed here.

Alongside FHE, two additional threat categories motivate the other two frontiers. First, the Harvest-Now-Decrypt-Later (HNDL) threat means that database transmissions encrypted today with Rivest–Shamir–Adleman (RSA) or Elliptic Curve Cryptography (ECC) can be retroactively decrypted once a sufficiently powerful quantum computer becomes available, making migration to NIST post-quantum lattice standards an urgent priority for all long-lived sensitive data [20], [21]. Second, the rapid proliferation of AI-powered attack tooling—including LLMs used to automate Structured Query Language (SQL) injection payload generation, database schema enumeration, and vulnerability exploit development—compresses the window between Common Vulnerabilities and Exposures (CVE) disclosure and working exploit code. The same capability that accelerates database attacks raises the question of whether LLMs can analogously accelerate

attacks on the lattice hard problems that protect both FHE ciphertexts and post-quantum key exchanges, which is the question addressed by the second frontier. Zero Trust Architecture, with its never-trust-always-verify enforcement of least-privilege access at every session boundary, provides the governance model that the third frontier operationalizes at the cryptographic level through functional encryption: each compute node is issued only the key material needed to evaluate its assigned function, and nothing more.

The following subsections provide the mathematical and systems background necessary to understand how lattice cryptography addresses each of these gaps.

3.1 *Lattice-based cryptography*

Lattice-based cryptography has grown from a theoretical curiosity into the dominant paradigm for post-quantum security over the past two decades. Its appeal rests on three properties: worst-case to average-case reductions that provide unusually strong security foundations, algebraic structure that enables rich functionalities such as fully homomorphic and functional encryption, and the absence of a known polynomial-time quantum algorithm for the underlying hard problems. We review the definitions and algorithms that appear throughout this survey.

3.1.1 *Lattices and hard problems*

An n-dimensional lattice $\Lambda \subset \mathbb{R}^n$ is the set of all integer linear combinations of n linearly independent basis vectors $\boldsymbol{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n]$:

$$\Lambda(\boldsymbol{B}) = \left\{ \sum_{i=1}^{n} z_i \boldsymbol{b}_i \mid z_i \in \mathbb{Z} \right\}.$$

The security of lattice-based cryptographic schemes rests on the conjectured hardness of several related problems.

Given a lattice basis $\boldsymbol{B}$, find a nonzero lattice vector $\boldsymbol{v} \in \Lambda(\boldsymbol{B})$ such that $\|\boldsymbol{v}\| \leq \gamma \times \lambda_l(\Lambda)$, where $\lambda_l(\Lambda)$ denotes the length of the shortest nonzero vector and $\gamma \geq 1$ is an approximation factor.

Given a lattice basis $\boldsymbol{B}$ and a target vector $\boldsymbol{t} \in \mathbb{R}^n$, find $\boldsymbol{v} \in \Lambda(\boldsymbol{B})$ such that $\|\boldsymbol{v} - \boldsymbol{t}\| \leq \gamma \times \mathrm{dist}(\boldsymbol{t}, \Lambda)$.

Both problems are NP-hard in the worst case for $\gamma = 1$. No algorithm is known that solves either in subexponential time for polynomial γ, and the best known algorithms run in time exponential in the lattice dimension; the hardness picture is surveyed in [14], and the worst-case to average-case connection that cryptography relies on is due to [22]. The phrasing here is the present authors' summary of that literature rather than a quotation from it.

3.1.2 *Fundamental parallelepiped*

The fundamental parallelepiped of basis $\boldsymbol{B}$ is the region

$$\mathcal{P}(\boldsymbol{B}) = \left\{ \sum_{i=1}^{n} x_i \boldsymbol{b}_i \mid x_i \in [0, 1) \right\}.$$

Every point in $\mathbb{R}^n$ lies in exactly one translate $\boldsymbol{v} + \mathcal{P}(\boldsymbol{B})$ for some $\boldsymbol{v} \in \Lambda(\boldsymbol{B})$, so the parallelepiped tiles $\mathbb{R}^n$ without overlap. The volume $\mathrm{vol}(\mathcal{P}(\boldsymbol{B})) = |\det(\boldsymbol{B})|$ is a lattice invariant that does not depend on which basis is chosen. Two bases $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ generate the same lattice if and only if $\boldsymbol{B}_1 = \boldsymbol{B}_2 \boldsymbol{U}$ for some unimodular integer matrix $\boldsymbol{U}$ with $|\det(\boldsymbol{U})| = 1$.

3.1.3 *Gram-Schmidt orthogonalization*

The Gram-Schmidt orthogonalization (GSO) of $\boldsymbol{B} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ produces a sequence of pairwise orthogonal vectors $\widetilde{\boldsymbol{b}}_i$ defined recursively by:

$$\widetilde{\boldsymbol{b}}_1 = \boldsymbol{b}_1, \quad \widetilde{\boldsymbol{b}}_i = \boldsymbol{b}_i - \sum_{j=1}^{i-1} \mu_{i,\,j}\, \widetilde{\boldsymbol{b}}_j, \quad \mu_{i,\,j} = \frac{\left\langle \boldsymbol{b}_i,\, \widetilde{\boldsymbol{b}}_j \right\rangle}{\left\| \widetilde{\boldsymbol{b}}_j \right\|^2}.$$

The GSO provides a tight lower bound on the shortest nonzero lattice vector: $\lambda_1(\Lambda) \geq \min_i \left\| \widetilde{\boldsymbol{b}}_i \right\|$. Basis quality is measured by how closely the log-norms $\log \left\| \widetilde{\boldsymbol{b}}_i \right\|$ follow a linearly decreasing profile (the Gaussian heuristic); the LLL and BKZ reduction algorithms aim to push bases toward this ideal shape, which is the foundation of the cryptanalytic attacks discussed in Section 5.

3.1.4 *Numerical example (GSO in $\mathbb{R}^2$)*

With $\boldsymbol{b}_1 = (3,\ 1)^\top$ and $\boldsymbol{b}_2 = (1,\ 2)^\top$ (Figure 3), set $\widetilde{\boldsymbol{b}}_1 = \boldsymbol{b}_1$. Then:

$$\mu_{2,\,1} = \frac{\langle \boldsymbol{b}_2,\, \boldsymbol{b}_1 \rangle}{\|\boldsymbol{b}_1\|^2} = \frac{3 \times 1 + 1 \times 2}{3^2 + 1^2} = \frac{5}{10} = \frac{1}{2}, \quad \widetilde{\boldsymbol{b}}_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix} - \frac{1}{2} \begin{pmatrix} 3 \\ 1 \end{pmatrix} = \begin{pmatrix} -1/2 \\ 3/2 \end{pmatrix}.$$

GSO norms: $\left\| \widetilde{\boldsymbol{b}}_1 \right\| = \sqrt{10} \approx 3.16$ and $\left\| \widetilde{\boldsymbol{b}}_2 \right\| = \sqrt{1/4 + 9/4} = \sqrt{5/2} \approx 1.58$. Lower bound: $\lambda_1 \geq \min(\sqrt{10},\ \sqrt{5/2}) = \sqrt{5/2} \approx 1.58$. The actual shortest vectors are $\boldsymbol{b}_2 = (1,\ 2)$ and $\boldsymbol{b}_2 - \boldsymbol{b}_1 = (-2,\ 1)$, each with length $\sqrt{5} \approx 2.24$, confirming the bound.

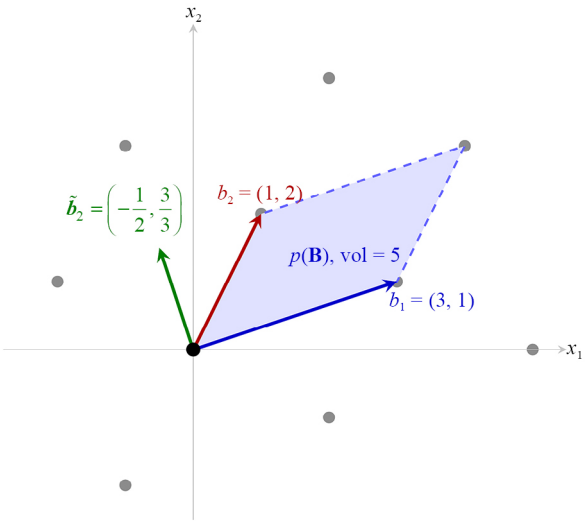

Figure 3. Two-dimensional lattice $\Lambda(B)$ with basis $b_1 = (3,\ 1)$, $b_2 = (1,\ 2)$ (black dots: lattice points; shaded region: fundamental parallelepiped $\mathcal{P}(B)$ with volume $|\det(B)| = 5$). The green vector $\widetilde{b}_2 = (-1/2,\ 3/2)$ is the second Gram-Schmidt orthogonalization vector, perpendicular to b_1. It gives the GSO lower bound $\lambda_1 \geq \left\| \widetilde{b}_2 \right\| = \sqrt{5/2} \approx 1.58$, confirmed by the actual shortest vectors $\pm b_2$ and $\pm(b_2 - b_1)$, each of length $\sqrt{5} \approx 2.24$

3.1.5 *Learning with errors*

The Learning with Errors (LWE) problem, introduced by Regev [6], is the primary algebraic vehicle for modern lattice cryptography.

Let $n,\ q \in \mathbb{Z}$ and let χ be an error distribution over $\mathbb{Z}_q$ (typically a discrete Gaussian with standard deviation σ). For a secret $\boldsymbol{s} \in \mathbb{Z}_q^n$, the LWE distribution $A_{s,\,\chi}$ produces samples $(\boldsymbol{a},\ b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ where $\boldsymbol{a} \leftarrow \mathbb{Z}_q^n$ uniformly and $b = \langle \boldsymbol{a},\ \boldsymbol{s} \rangle + e (\bmod q)$ with $e \leftarrow \chi$.

Given m samples, decide whether they are drawn from $A_{s,\,\chi}$ for some fixed $\boldsymbol{s}$, or from the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

Regev showed a quantum reduction from worst-case SVP on n-dimensional lattices to solving LWE with n samples [6]. The ring variant RLWE [7] operates over polynomial rings $R_q = \mathbb{Z}_q[x]/(f(x))$ and enables more efficient schemes with quasi-linear operations.

3.1.6 *Search-LWE versus decision-LWE*

Search-LWE asks to recover $\boldsymbol{s}$ from polynomially many LWE samples; Decision-LWE asks to distinguish LWE samples from the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. A polynomial-time reduction shows the two are equivalent [6]: a Decision-LWE oracle can be used to recover $\boldsymbol{s}$ one coordinate at a time by testing candidate values against the distinguisher. This mirrors the CDH-to-DDH equivalence in classical cryptography, with the additional property that no efficient quantum algorithm is known for either variant. A function $\mu(n)$ is negligible if it decreases faster than any inverse polynomial; LWE hardness means that every Probabilistic Polynomial-Time (PPT) adversary achieves distinguishing advantage at most $\mu(n)$ for some negligible μ.

3.1.7 *Numerical example (LWE sample and Regev encryption)*

Let $n = 4,\ q = 97,\ \sigma = 1$, and secret $\boldsymbol{s} = (3,\ 2,\ 5,\ 4)^\top \in \mathbb{Z}_{97}^4$. Sample $\boldsymbol{a} = (22,\ 7,\ 11,\ 43)^\top$ uniformly at random and draw noise $e = 2$ from the discrete Gaussian. Compute the noisy inner product:

$$\langle \mathbf{a},\ \mathbf{s} \rangle = 22 \times 3 + 7 \times 2 + 11 \times 5 + 43 \times 4 = 66 + 14 + 55 + 172 = 307,$$

$$b = 307 \bmod 97 + e = 16 + 2 = 18.$$

The pair $(\boldsymbol{a},\ b) = ((22,\ 7,\ 11,\ 43),\ 18)$ is the LWE sample. An adversary seeing this pair must distinguish $b = 18$ from a uniformly random element of $\mathbb{Z}_{97}$; by Decision-LWE hardness, no efficient algorithm can do so with non-negligible advantage.

Encryption of bit $\mu = 1$ (Regev scheme, threshold $\lfloor q/2 \rfloor = 48$):

$$\boldsymbol{u} = \boldsymbol{a} = (22,\ 7,\ 11,\ 43)^\top, \quad v = b + \mu \times 48 = 18 + 48 = 66.$$

Decryption with secret $\boldsymbol{s}$:

$$v - \langle \boldsymbol{s},\ \boldsymbol{u} \rangle \equiv 66 - 307 \ (\bmod 97)$$

$$\equiv 66 - 16 = 50 \ (\bmod 97)$$

$$\approx 48 \Longrightarrow \mu = 1.$$

The residual $50 - 48 = 2 = e$ recovers the original noise and confirms correctness: the error is small enough that rounding to the nearest encoding succeeds.

Table 3. LWE security estimates for representative parameter sets. Security bits are estimated via the BKZ cost model calibrated to Kyber-512 and Kyber-1024 target levels; see Figure 4. Toy parameters in the last row are for pedagogical illustration only

n	q	$\log_2 q$	σ	Estimated security
512	3,329	11.7	3.2	≈ 114 bits
1,024	3,329	11.7	3.2	≈ 236 bits
2,048	3,329	11.7	3.2	≈ 480 bits
4	97	6.6	1.0	$\ll 10$ bits (toy)

Table 3 shows how the estimate scales with the lattice dimension. Production parameters use $n \geq 512$ and $q \approx 3,329$, as in CRYSTALS-Kyber, standardized as FIPS 203 [4]. The simplified cost model used for the table gives roughly 114 bits at $n = 512$, short of 128; ML-KEM-512 is specified against NIST security category 1 rather than against a bit count, and its actual margin depends on the module structure and on estimator choices that the coarse model here does not capture. The table is intended to show how the estimate scales with n, not to certify a standardized parameter set. The toy parameters above $(n = 4,\ q = 97)$ provide essentially no security but illustrate the algebraic structure that scales to cryptographic strength.

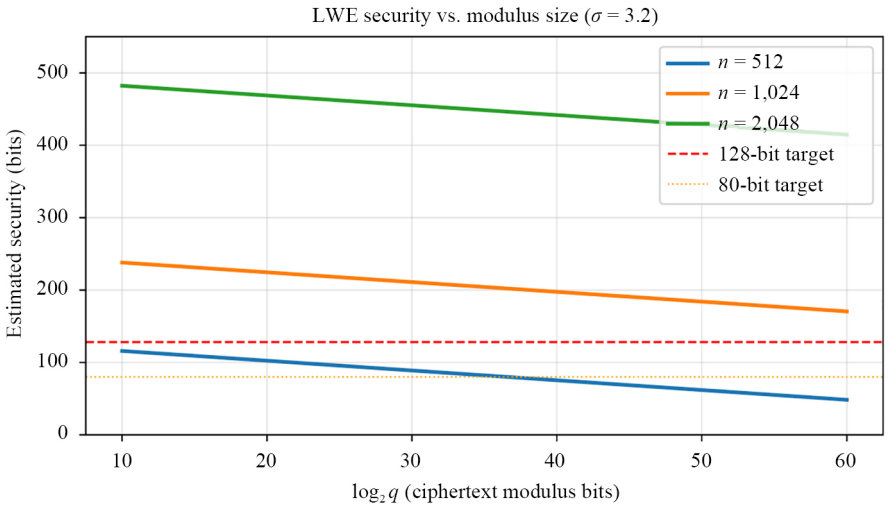

Figure 4. Estimated LWE security in bits as a function of modulus size $\log_2 q$ for three LWE dimensions $n \in \{512,\ 1,024,\ 2,048\}$ with $\sigma = 3.2$. Larger moduli reduce security; NIST-standardized parameter sets target at least 128 bits. The figure also illustrates the security-efficiency tradeoff: wider ciphertext moduli (needed for FHE depth) require larger n to maintain security

3.1.8 *Fully homomorphic encryption*

The ability to compute on encrypted data without decryption has been called the "holy grail" of cryptography, and LWE-based constructions have made it practical. We briefly introduce the four families of FHE schemes that appear in Section 4.

Gentry's seminal 2009 construction [8] showed that it is possible to evaluate arbitrary Boolean circuits on ciphertexts without decrypting, using the concept of bootstrapping to refresh noisy ciphertexts. Modern FHE schemes fall into four families, summarized in Table 4.

Table 4. Major FHE scheme families and their properties relevant to LLM workloads

Scheme	Hard problem	Supported ops	Noise type	LLM fit
BGV [9]	RLWE	Integer arithmetic (leveled)	Additive	Discrete activations
BFV [23]	RLWE	Integer arithmetic	Additive	Discrete activations
CKKS [10]	RLWE	Approx. floating-point	Approximate	Attention, Gaussian Error Linear Unit (GELU)
TFHE [24]	LWE/TLWE	Bit-by-bit (bootstrapped)	Discrete	Arbitrary Boolean

The FHE noise budget is central to the security-correctness tradeoff. Each homomorphic operation consumes noise budget; bootstrapping restores it at significant computational cost. CKKS is particularly relevant to LLM inference because it supports approximate arithmetic over packed complex vectors, mapping naturally to floating-point tensor operations.

3.1.9 *Functional encryption*

Functional encryption introduces a fundamental shift from "decrypt all or nothing" to "compute and reveal only what is authorized." It is the theoretical foundation for the decentralized inference architecture studied in Section 6.

Functional encryption [11] generalizes public-key encryption: a functional decryption key sk_f enables computing $f(m)$ from a ciphertext $\mathrm{Enc}(\mathrm{pk}, m)$ without revealing m or anything beyond $f(m)$.

A functional encryption scheme for a function family $\mathcal{F}$ over message space $\mathcal{M}$ is a tuple (KeyGen, Enc, Dec, FKGen) satisfying:

• Correctness: For all $f \in \mathcal{F}$, $m \in \mathcal{M}$: $\mathrm{Dec}(\mathrm{FKGen}(\mathrm{msk}, f), \mathrm{Enc}(\mathrm{pk}, m)) = f(m)$.

• Simulation security: A distinguisher that sees $\mathrm{ct} = \mathrm{Enc}(\mathrm{pk}, m)$ and arbitrary functional keys learns nothing about m beyond $\{f_i(m)\}$.

Lattice-based inner-product FE schemes [25] support the function class $f_{\boldsymbol{y}}(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$, which is linear in the encrypted plaintext. This class maps directly to the linear projection layers of a transformer, where each output neuron is an inner product of a public weight row with the encrypted activation. It does not extend to the attention mechanism, which is bilinear in two encrypted activations and therefore lies outside the innerproduct functionality class; Section 6.3.1 states this limitation precisely and proves why no combination of inner-product keys evaluates it.

3.1.10 *Lattice reduction algorithms*

Lattice attacks on LWE and related problems all reduce to finding short vectors in carefully constructed lattices. Two algorithms form the practical backbone of this enterprise, and their interaction is the subject of Section 5.

3.1.11 *LLL Algorithm*

Lenstra et al. [26] showed that a reduced basis with approximation factor $\gamma = 2^{(n-1)/2}$ can be computed in polynomial time via orthogonalization and size-reduction swaps.

3.1.12 *BKZ Algorithm*

The Block Korkine-Zolotarev (BKZ) algorithm, introduced by Schnorr and Euchner [27], generalizes LLL through blockwise reduction of projected sublattices. (The name derives from the 1,873 work of Korkine and Zolotarev on sphere packings; BKZ is unrelated to lattice sieving, which is a separate algorithm used as the SVP oracle inside BKZ .) BKZ_β calls an SVP oracle on projected blocks of dimension β, achieving $\gamma \approx \beta^{n/\beta}$ with time complexity roughly $2^{c\beta}$ for constant c. The concrete hardness of LWE is measured by the smallest β required to solve the associated lattice problem.

3.1.13 *NIST post-quantum cryptography standardization*

The standardization of post-quantum cryptographic algorithms by the National Institute of Standards and Technology (NIST) is the primary policy driver for the transition away from RSA and ECC [21]. Because RSA and ECC security rests on the hardness of integer factorization and discrete logarithm problems that Shor's algorithm solves in polynomial quantum time, any organization that relies on these primitives faces an existential threat from sufficiently large quantum hardware [4], [20].

3.1.14 *First-wave standards (2024)*

In August 2024, NIST finalized three post-quantum cryptographic standards. Two are built on hard lattice problems; the third is a hash-based backup:

• FIPS 203 (ML-KEM) [4]: Module-Lattice-Based Key-Encapsulation Mechanism, derived from CRYSTALS-Kyber. Its security rests on the Module-LWE problem over the ring $R_q = \mathbb{Z}_q[X]/\left(X^{256}+1\right)$ with $q = 3,329$; the three parameter sets ML-KEM-512, ML-KEM-768, and ML-KEM1024 correspond to module ranks $k = 2,\ 3,\ 4$ and to NIST security categories 1, 3, and 5 respectively. The polynomial degree is fixed at $n = 256$ across all three sets, and security is scaled by increasing k rather than n.

• FIPS 204 (ML-DSA) [18]: Module-Lattice-Based Digital Signature Algorithm, derived from CRYSTALS-Dilithium. Also Module-LWE based, with the additional Module-SIS assumption required for unforgeability.

• FIPS 205 (SLH-DSA) [19]: Stateless Hash-Based Digital Signature Algorithm, derived from SPHINCS+. Provides a hash-based (non-lattice) backup signature standard whose security depends only on the properties of the underlying hash function.

A common and consequential imprecision is to describe ML-KEM as RLWE-based. It is not. Module-LWE interpolates between plain LWE ($k = n$, degree 1) and RLWE ($k = 1$, degree n), and the module structure is a deliberate engineering choice: it allows a single optimized number-theoretic transform for a fixed degree-256 ring to be reused across all security levels, with security tuned by the module rank. This distinction matters for the present survey because the FHE and FE constructions discussed below sit at different points on the same spectrum, as Section 3.3 makes precise.

3.1.15 *Second-wave candidates*

On 11 March 2025, NIST selected a fifth algorithm, Hamming Quasi-Cyclic (HQC), as a code-based encryption backup to ML-KEM [28]. HQC is not a replacement: ML-KEM remains the primary key-encapsulation mechanism, and HQC exists so that a future break of the module-lattice assumption would not leave general-purpose encryption without a standardized option. A draft HQC standard was scheduled for public comment approximately one year after the announcement, with finalization expected in 2027. Separately, NIST has advanced a secondary digital signature track that prioritizes security assumptions diverse from lattice problems. This diversification hedges against the possibility that a future algorithmic breakthrough could undermine all lattice-based security simultaneously, a systemic risk that is particularly acute for the present survey, since all three frontiers examined here rest on the same family of assumptions.

3.1.16 *Migration deadlines and the harvest-now-decrypt-later threat*

Migration guidance is driven by the Harvest-Now-Decrypt-Later (HNDL) threat model. NIST IR 8547 [29] sets out a proposed transition trajectory in which public-key algorithms providing roughly 112 bits of classical security, such as RSA-2048 and ECC P-256, are targeted for migration after 2030 and eventual retirement after 2035. Deprecation and disallowance are distinct regulatory states, and the distinction is often lost in secondary reporting: a deprecated algorithm may still be used for legacy interoperability, whereas a disallowed algorithm may not be used for the stated purpose at all. Under HNDL, adversaries intercept and archive encrypted ciphertext today, intending to decrypt it retroactively once quantum hardware matures. Data with long classification lifetimes (medical records, national security communications, AI model weights) is particularly vulnerable [20].

3.1.17 *NIST, AI supply chains, and funding*

NIST does not develop AI models directly, but plays three intersecting roles in the LBC-LLM landscape:

1. AI Supply Chain Security. NIST workshops and risk profiles (including the NIST AI Risk Management Framework [30] and the NIST Quantum Profile) help organizations address the dual threat of quantum computing and adversarial AI, covering secure supply chains, model provenance, and geopolitical risk.

2. Protecting AI Assets. NIST-approved post-quantum algorithms are used by organizations to encrypt intellectual property, LLM model weights, and sensitive training data against quantum-capable adversaries [4], [20].

Table 5 summarizes the key NIST PQC milestones relevant to this survey.

Table 5. NIST post-quantum cryptography standardization timeline

Year	Milestone	Relevance to LBC-LLM
2016	NIST PQC competition launched	Established structured lattices as the dominant hard problem family
2022	Four algorithms selected (draft)	Kyber, Dilithium, Falcon, SPHINCS+
Aug. 2024	FIPS 203, 204, 205 finalized	Module-lattice standards ML-KEM and ML-DSA; hash-based SLH-DSA
Nov. 2024	NIST IR 8547 draft released [29]	Sets deprecation and disallowance trajectory
Mar. 2025	HQC selected as fifth algorithm	Code-based diversity hedge against a module-lattice break
2030	Classical public-key deprecated	HNDL exposure window closes for new systems
2035	Classical public-key disallowed	Terminal deadline for quantum-vulnerable primitives

3.2 *One assumption, three capabilities: the algebraic bridge*

Systems engineers encountering this literature reasonably ask why key exchange, homomorphic encryption, and functional encryption, which serve entirely different purposes, are repeatedly said to share a foundation. The answer is that all three are instances of a single template, and the differences between them are choices about what to do with one structural fact. This subsection states that template without proofs; it is intended as an orientation for readers who will not work through the constructions in detail.

3.2.1 *The common template*

Table 6 sets the three capabilities side by side under a single parameterization of the same LWE relation.

Table 6. The same LWE relation, parameterized three ways. The differences in modulus and degree are consequences of the number of homomorphic operations each capability must support before decryption, not of differing security targets; all three rows are calibrated to roughly 128-bit security

Capability	Assumption	Degree n	Modulus q	Ops
ML-KEM (KEM)	Module-LWE, rank k	256	3329	0
CKKS/BFV (FHE)	RLWE, rank 1	$2^{13}-2^{17}$	$2^{200}-2^{800}$	10^2-10^4
ALS (IPFE)	Plain LWE	n/a (matrix)	$2^{30}-2^{60}$	1

Every scheme discussed in this review is built from the LWE relation

$$b = \langle \boldsymbol{a},\ \boldsymbol{s} \rangle + e (\bmod q),$$

in which $\boldsymbol{a}$ is public and uniformly random, $\boldsymbol{s}$ is the secret, and e is a small error term. Two properties of this relation do all the work. First, it is pseudorandom: without $\boldsymbol{s}$, the pair $(\boldsymbol{a},\ b)$ is computationally indistinguishable from uniform,

which is what makes it an encryption. Second, it is linear up to the error: adding two such relations yields another valid relation with error $e_1 + e_2$. Encryption, homomorphism, and functional decryption are three different exploitations of that second property.

3.2.2 *Where the three capabilities diverge*

3.2.2.1 *Key encapsulation (ML-KEM)*

The goal is to transmit one short secret and then stop. Only a single decryption is ever performed, so the error never needs to remain small across operations. This permits an aggressively small modulus ($q = 3,329$) and a fixed small degree ($n = 256$), with security scaled by the module rank k. The design is optimized for bandwidth and speed, and it deliberately sacrifices any capacity for computation on ciphertexts.

3.2.2.2 *Homomorphic encryption (CKKS, BFV, BGV)*

The goal is to perform many operations before decrypting. Multiplication of two ciphertexts multiplies their errors, so the error grows and must be budgeted. Two changes follow directly. The modulus must be very large, on the order of 2^{200} to 2^{800}, to leave room for that growth; and the ring degree must be large, typically 2^{13} to 2^{17}, to preserve security against a modulus that large. This is why an FHE ciphertext is measured in megabytes while an ML-KEM ciphertext is measured in hundreds of bytes: the size is the price of the computation budget, not an implementation inefficiency.

3.2.2.3 *Functional encryption (ALS inner-product)*

The goal is to release exactly one linear function of the plaintext and nothing else. Here the key insight is that a functional key can itself be a linear combination of secrets, $\mathrm{sk}_{\mathrm{y}} = \boldsymbol{Z}^{\top}\boldsymbol{y}$, so that decrypting with it collapses the ciphertext directly to $\langle \boldsymbol{x},\ \boldsymbol{y} \rangle$ without ever revealing $\boldsymbol{x}$. Because only one operation is performed, the error budget is modest, but the scheme pays elsewhere: the ALS construction operates over plain LWE with matrix dimensions proportional to the plaintext length, which is the origin of the key-size problem analysed in Section 6.8.

3.2.3 *Consequences of the shared foundation*

Three consequences follow, and they are the reason this survey treats the three frontiers together rather than separately.

First, a single cryptanalytic advance propagates. An improvement in lattice reduction that reduces the effective security of ML-KEM reduces the security of every CKKS and ALS deployment calibrated against the same estimator. Parameter selection across all three is downstream of one literature, which is why Section 5 is relevant to engineers who have no interest in cryptanalysis as such.

Second, hardware investment is shared. The number-theoretic transform is the dominant cost in all three settings. An NTT accelerator built for post-quantum key exchange is architecturally close to one needed for homomorphic inference, which materially improves the economics of the hardware argument in Section 9.

Third, the assumptions are correlated, and this is a risk rather than a benefit. An organization that adopts ML-KEM for transport security, CKKS for encrypted analytics, and lattice FE for access control has not diversified. It has concentrated its exposure in one problem family. The NIST decision to standardize the code-based HQC alongside ML-KEM reflects exactly this concern at the level of national infrastructure, and the same reasoning applies to system design.

3.3 *Large language models*

We briefly review the architectural features of LLMs that are most relevant to the three frontiers: the linear algebra of transformer layers, which maps cleanly to IPFE; the training dynamics, which determine the noise tolerance of FHE-based protocols; and the empirically demonstrated ability of sufficiently large models to act as general-purpose reasoning engines, which motivates the LLM-guided cryptanalysis frontier.

3.3.1 *Architecture*

Modern LLMs are transformer-based neural networks [31] consisting of stacked self-attention layers and feed-forward networks. For an input token sequence $(t_1, \ldots, t_L)$, the l-th transformer block computes:

$$\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} = \boldsymbol{X}_l \boldsymbol{W}_Q, \boldsymbol{X}_l \boldsymbol{W}_K, \boldsymbol{X}_l \boldsymbol{W}_V$$

$$\boldsymbol{A}_l = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_k}}\right)\boldsymbol{V}$$

$$\boldsymbol{X}_{l+1} = \text{LayerNorm}\left(\boldsymbol{A}_l + \text{FFN}(\boldsymbol{A}_l)\right),$$

where $\boldsymbol{W}_Q$, $\boldsymbol{W}_K$, $\boldsymbol{W}_V$ are learned weight matrices and FFN is a two-layer feed-forward network with a nonlinear activation (GELU or SwiGLU).

3.3.2 *Privacy vulnerabilities*

LLMs trained on sensitive data exhibit a set of privacy vulnerabilities that motivate the cryptographic protections surveyed in this paper. Three classes are directly relevant, and each maps onto one of the frontiers: training-data leakage, which Frontier 1 addresses through encrypted training; the integrity of the cryptographic assumptions themselves, which Frontier 2 probes; and disclosure of intermediate state during inference, which Frontier 3 restricts through functional encryption. Deployment-time attack surfaces such as prompt injection and model poisoning are outside the confidentiality scope of this review, as stated in Section 2.3. The three vulnerability classes below are the specific mechanisms each frontier must contend with.

3.3.3 *Memorization*

Carlini et al. [32] demonstrated that LLMs verbatim memorize and reproduce training sequences, including sequences appearing in a single document. A subsequent systematic study [33] quantifies the effect, reporting log-linear growth of memorization in model capacity, in the number of times a sequence is duplicated in the training corpus, and in the length of the prompting context.

3.3.4 *Membership inference*

Shokri et al. [34] showed that an adversary can infer whether a specific record was in the training set by comparing the model's output confidence on the target record versus a shadow-trained model.

3.3.5 *Model inversion*

An adversary with API access can reconstruct approximate training inputs using model inversion techniques [35], which exploit output confidence information through gradient-based optimization or repeated querying.

3.3.6 *Parameter-efficient fine-tuning*

Full fine-tuning of LLMs with billions of parameters becomes less feasible as model size grows, and the cost that Hu et al. [36] identify as prohibitive is specifically that of storing and switching between independently fine-tuned copies of the full model, one per downstream task. Low-Rank Adaptation (LoRA) [36] inserts trainable low-rank decomposition matrices $\Delta\boldsymbol{W} = \boldsymbol{B}\boldsymbol{A}$ ($\boldsymbol{B} \in \mathbb{R}^{d\times r}$, $\boldsymbol{A} \in \mathbb{R}^{r\times k}$, $r \ll \min(d, k)$) in parallel with frozen pretrained weights. For GPT-3 175B the authors report a ten-thousand-fold reduction in trainable parameters and a threefold reduction in GPU memory, with

the per-task checkpoint falling from 350 GB to 35 MB. The adapter is the natural integration point for FHE-based private fine-tuning, since it is the only part of the network whose weights change.

3.3.7 *LLMs as heuristic agents*

Beyond text generation, LLMs have been successfully used as black-box optimizers and search heuristics via in-context learning [3]. When presented with a structured description of a combinatorial problem, sufficiently large LLMs can generate plausible candidate solutions, evaluate partial progress, and propose adaptive strategies, properties that have been exploited in algorithm configuration and hyperparameter search. Section 5 examines how these capabilities apply to lattice reduction.

4. Fully homomorphic encryption for private LLM training

The first frontier concerns a fundamental tension in modern AI deployment: organizations wish to fine-tune powerful LLMs on proprietary or sensitive corpora, but doing so on third-party cloud infrastructure exposes training data to the server. FHE offers a cryptographic path toward resolution by allowing the server to execute gradient updates directly on ciphertexts, learning only an encrypted model and nothing about the underlying data; practical deployment at scale remains a research goal rather than an established capability. This section formalizes the setting, surveys the state of the art for each compatible FHE scheme, and analyzes the relationship between FHE noise and differential privacy.

4.1 *Problem formulation*

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ be a sensitive training corpus and let θ be the parameters of an LLM. Standard fine-tuning solves:

$$\theta^* = \operatorname*{argmin}_{\theta} \frac{1}{N} \sum_{i=1}^{N} \ell(\theta; x_i, y_i),$$

in plaintext, exposing $\mathcal{D}$ to any party who controls the training infrastructure. An FHE-based training protocol replaces this with:

$$\mathrm{Eval}(\mathrm{KeyGen}(\lambda), \mathrm{TrainStep}, \mathrm{Enc}(\mathrm{pk}, x_i), \mathrm{Enc}(\mathrm{pk}, y_i), \mathrm{Enc}(\mathrm{pk}, \theta)) \to \mathrm{Enc}(\mathrm{pk}, \theta^*),$$

where the evaluation is performed on ciphertexts and the server learns nothing about $\mathcal{D}$ or θ^* beyond what is revealed by $\mathrm{Dec}(\mathrm{sk}, \cdot)$.

Figure 5 illustrates the complete pipeline, including the privacy auditing loop. After the model owner decrypts θ^*, a standard Membership Inference Attack (MIA) or canary extraction test probes the plaintext weights for residual memorization. This auditing step is independent of the FHE protocol and quantifies empirically whether the combination of FHE noise and explicit DP-SGD is sufficient to meet the organization's privacy policy.

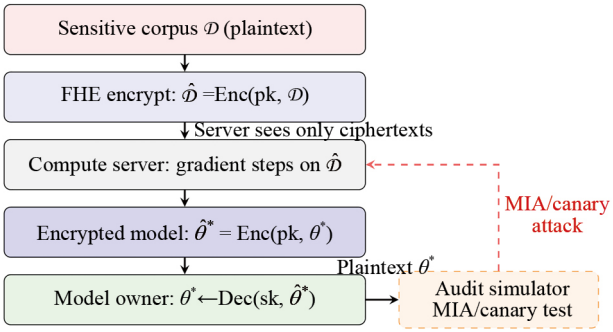

Figure 5. FHE-based private LLM training and auditing pipeline. The compute server operates exclusively on ciphertexts, returning an encrypted model. After decryption by the model owner, a membership inference attack or canary extraction test audits residual memorization in θ^*. The dashed arrow represents the adversarial threat model used to measure empirical privacy leakage; its goal is to verify that FHE noise (complemented by explicit DP-SGD [37] if needed) is sufficient to prevent reconstruction of individual training samples

4.2 *FHE schemes and LLM compatibility*

Not all FHE schemes are equally suited to LLM workloads. The choice of scheme governs which operations can be executed natively, how large the circuit depth budget is before bootstrapping, and what precision is available for gradient accumulation. The central challenge of FHE-based LLM training is the non-polynomial nature of transformer operations. The softmax, GELU, and LayerNorm operations require either polynomial approximation or circuit-based decomposition. Prior applied work has demonstrated this challenge concretely in security analytics, where CKKS polynomial approximation of the Sigmoid activation has been applied to encrypted log anomaly detection [12], and where probabilistic polynomial approximation techniques have been shown to achieve improved efficiency over Chebyshev interpolation for selected Sign and Compare primitives under the evaluated FHE settings [38]. Those primitives are the building blocks for the comparison-based operations required by ReLU-family activations and by the gradient clipping discussed in Section 4.4. More recently, THOR [39] and NEXUS [40] have advanced transformer-scale encrypted inference by supporting computationally expensive components such as softmax, LayerNorm, and GELU, which is the state of the art surveyed in Section 4.6. Table 7 maps each LLM operation to the most compatible FHE scheme.

Table 7. Mapping of LLM operations to FHE scheme suitability

LLM operation	Mathematical form	FHE approach	Best scheme
Linear projection	$\boldsymbol{Wx}$	Direct matrix-vector HMult	CKKS/BFV
Attention score	$\boldsymbol{q}^\top \boldsymbol{k}/\sqrt{d}$	HMult + HRot	CKKS
Softmax	$e^{x_i}/\sum_j e^{x_j}$	Polynomial approx. of degree ≥ 7	CKKS
GELU	$x \times \Phi(x)$	Chebyshev/probabilistic polynomial approx. [38], [39]	CKKS
LayerNorm	$(x-\mu)/\sigma$	Iterative variance estimation	CKKS
LoRA update	$\boldsymbol{BAx}$	Low-rank HMult (cheap)	CKKS/BGV

4.2.1 *CKKS for floating-point tensors*

The CKKS scheme [10] encodes a vector of $N/2$ complex numbers into a single ciphertext via the canonical embedding of the cyclotomic ring $R_q = \mathbb{Z}[x]/\left(x^N+1\right)$. Homomorphic operations introduce a controlled approximation error bounded by 2^{-p} where p is the message precision. This matches the tolerance for floating-point roundoff in LLM training with mixed-precision arithmetic.

4.2.2 *TFHE for exact boolean circuits*

TFHE [24] supports gate-by-gate bootstrapping on LWE ciphertexts over the torus $\mathbb{T} = \mathbb{R}/\mathbb{Z}$, achieving submillisecond per-gate evaluation. For LLMs, TFHE is better suited for discrete operations such as token embedding lookups or quantized activation functions, where exact arithmetic is required.

4.2.3 *FHE-based fine-tuning protocols*

Given the scheme-operation mapping above, the question is how to structure a training protocol that minimizes homomorphic depth while preserving the security and convergence properties of standard finetuning. Full homomorphic training of billion-parameter LLMs remains beyond current FHE efficiency. Two protocol architectures are theoretically coherent and adaptable to existing encrypted inference frameworks, and are presented here as research proposals rather than deployed systems.

4.2.4 *Encrypted LoRA fine-tuning*

LoRA limits trainable parameters to low-rank matrices $\{\boldsymbol{A}_l,\ \boldsymbol{B}_l\}$ per layer l. An encrypted LoRA protocol proceeds as follows:

Algorithm 1:

Require: Frozen pretrained weights θ_0, training data $\mathcal{D}$, rank r, FHE context ctx

Ensure: Encrypted LoRA weights $\mathrm{Enc}(\mathsf{pk},\ \{\boldsymbol{A}_l,\ \boldsymbol{B}_l\})$

$(\mathsf{pk},\ \mathsf{sk}) \leftarrow \mathrm{KeyGen}(\mathsf{ctx})$

Initialize $\boldsymbol{A}_l \leftarrow \mathcal{N}(0,\ \sigma^2)$, $\boldsymbol{B}_l \leftarrow 0$ for each layer l

for each batch $(X,\ Y) \in \mathcal{D}$ **do**

  $\hat{X} \leftarrow \mathrm{Enc}(\mathsf{pk},\ X)$

  $\hat{Y} \leftarrow \mathrm{Enc}(\mathsf{pk},\ Y)$

  $\{\hat{H}_l\} \leftarrow \mathrm{Eval}(\mathsf{ctx},\ \mathrm{Forward},\ \hat{X},\ \theta_0,\ \{\mathrm{Enc}(\boldsymbol{A}_l)\},\ \{\mathrm{Enc}(\boldsymbol{B}_l)\})$ ▷ Cache per-layer activations for reuse in the backward pass

  $\hat{L} \leftarrow \mathrm{Eval}(\mathsf{ctx},\ \mathrm{CrossEntropy},\ \hat{H}_L,\ \hat{Y})$ ▷ Requires polynomial softmax

  $\mathrm{Enc}(\delta_L) \leftarrow \mathrm{Eval}(\mathsf{ctx},\ \partial\,\mathrm{CrossEntropy},\ \hat{H}_L,\ \hat{Y})$

  for $l = L$ **down to** 1 **do**

    $\mathrm{Enc}(\delta_{l-1}) \leftarrow \mathrm{Eval}(\mathrm{Backprop}_l,\ \mathrm{Enc}(\delta_l),\ \theta_0,\ \mathrm{Enc}(\boldsymbol{A}_l),\ \mathrm{Enc}(\boldsymbol{B}_l))$

    $\mathrm{Enc}(\nabla B_l) \leftarrow \mathrm{Enc}(\delta_l) \times \mathrm{Enc}(\boldsymbol{A}_l \hat{H}_{l-1})^{\mathsf{T}}$

    $\mathrm{Enc}(\nabla \boldsymbol{A}_l) \leftarrow \mathrm{Enc}(\boldsymbol{B}_l)^{\mathsf{T}} \times \mathrm{Enc}(\delta_l) \times \hat{H}_{l-1}^{\mathsf{T}}$

  end for

  for $l = 1$ **to** L **do**

    $\mathrm{Enc}(\boldsymbol{A}_l) \leftarrow \mathrm{Enc}(\boldsymbol{A}_l) - \eta \times \mathrm{Enc}(\nabla \boldsymbol{A}_l)$ ▷ Plaintext-scalar multiplication

    $\mathrm{Enc}(\boldsymbol{B}_l) \leftarrow \mathrm{Enc}(\boldsymbol{B}_l) - \eta \times \mathrm{Enc}(\boldsymbol{B}_l)$ ▷ Both factors updated; see remark below

  end for

  if $\mathrm{level}(\mathrm{Enc}(\boldsymbol{A}_l)) < \tau$ **then**

    $\mathrm{Enc}(\boldsymbol{A}_l),\ \mathrm{Enc}(B_l) \leftarrow \mathrm{Bootstrap}(-)$ ▷ Unavoidable: depth grows without bound over steps

  end if

end for

return $\{\mathrm{Enc}(\mathsf{pk},\ \boldsymbol{A}_l),\ \mathrm{Enc}(\mathsf{pk},\ \boldsymbol{B}_l)\}$

Three points about Algorithm 1 require emphasis, because an earlier formulation of this protocol was materially incomplete.

First, both LoRA factors must be updated. The adaptation is $\Delta W_l = \boldsymbol{B}_l \boldsymbol{A}_l$, and freezing $\boldsymbol{B}_l$ at its zero initialization while updating only $\boldsymbol{A}_l$ leaves $\Delta W_l = 0$ for all time; the model would not train at all. This matters cryptographically as

well as numerically: the product $\boldsymbol{B}_l\boldsymbol{A}_l$ is a ciphertext-ciphertext multiplication once both factors are encrypted, which consumes a multiplicative level per layer per step and is the dominant driver of the bootstrapping frequency τ.

Second, the backward pass is not a single homomorphic evaluation. It is a sequential traversal in which each layer's error signal δ_l depends on the layer above, so the multiplicative depth of one training step scales with network depth L rather than being constant. For a 12-layer encoder with ciphertext-ciphertext products at each layer this exhausts a typical CKKS level budget within a single step, making bootstrapping a per-step rather than a per-epoch cost. This is the single largest obstacle to the protocol and is not resolved here.

Third, the gradient expression $\nabla_{\boldsymbol{A}_l} = \boldsymbol{B}_l^\top \delta_l \boldsymbol{H}_{l-1}^\top$ does reduce to matrix products that CKKS supports via rotation-and-sum algorithms [10], but the required rotation keys scale with the matrix dimensions, and the cached activations $\hat{H}_l$ must be retained in ciphertext across the forward and backward passes. For a hidden dimension of 768 and a batch of 32, this working set alone is on the order of several gigabytes at typical CKKS parameters, before any optimizer state.

4.2.5 *Unresolved circuit and systems requirements*

Algorithm 1 is a protocol sketch, not an implementable specification. The following components are required for it to run and are either unsolved or carry costs that have not been measured at transformer scale. They are listed explicitly so that the gap between this proposal and a working system is not understated.

• **Softmax.** Attention requires softmax, which involves exponentiation and division by a data-dependent sum. Neither is natively supported. Polynomial approximation requires a known input range, which is not available under encryption, and Goldschmidt division for the normalization adds further depth.

• **Cross-entropy loss.** Requires a logarithm, approximable only over a bounded interval, and the same normalization problem as softmax.

• **LayerNorm.** Requires an inverse square root of a data-dependent variance. Iterative methods converge only from a good initial estimate, which again presumes range knowledge unavailable in ciphertext.

• **Activations.** GELU and SiLU are non-polynomial. Minimax polynomial approximations exist but their accuracy degrades outside the fitted range, and error compounds across layers in a manner that has not been characterized for training, as opposed to inference.

• **Ciphertext packing and rescaling.** The packing layout that is optimal for the forward pass is generally not optimal for the transposed products in the backward pass, forcing either repacking, which costs rotations, or a suboptimal layout throughout.

• **Bootstrapping cadence.** Depth grows with both network depth and step count. A bootstrapping schedule that keeps the noise budget positive without dominating runtime has not been demonstrated for a training workload.

• **Optimizer state.** Adam requires elementwise division and square roots over running moment estimates, all under encryption. Plain SGD avoids this but converges more slowly, which increases the number of encrypted steps and therefore the total bootstrapping cost.

• **Numerical precision.** CKKS is approximate. Whether the accumulated approximation error over thousands of steps permits convergence to a useful model is an open empirical question; no convergence result for encrypted transformer fine-tuning has been published.

• **Differentially private clipping.** As established in Section 4.4, obtaining a post-decryption privacy guarantee requires per-example gradient clipping, which is a comparison against a norm and is among the most expensive CKKS primitives.

The honest summary is that encrypted LoRA fine-tuning is at present a research direction with a plausible structure and no demonstrated instance. The hybrid protocol below is more tractable precisely because it relocates every item in the list above to the plaintext domain.

4.2.6 *Hybrid two-party protocol*

An alternative is a hybrid protocol where the data owner holds the secret key and the compute server holds the encrypted data:

1. Offline phase: Data owner encrypts D and LoRA initialization under pk and sends to server.

2. Compute phase: Server runs encrypted forward and backward passes, producing encrypted gradients. For non-polynomial activations, the server may send garbled circuit inputs to a helper or use an approximate polynomial.

3. Decrypt phase: Data owner decrypts the gradient updates, aggregates, and re-encrypts updated LoRA weights.

This mirrors the CrypTen [41] and Delphi [42] frameworks but extends them to the fine-tuning setting. The same division of labour, with linear algebra evaluated homomorphically and non-linearities handled outside the homomorphic domain, underlies the interactive transformer protocols of Iron [43] and PUMA [44], and the practical limit of the approach is set by the number of client-server rounds it requires.

4.3 *Ciphertext error is not differential privacy*

An appealing but incorrect argument circulates in the applied literature: FHE adds Gaussian noise, differential privacy adds Gaussian noise, therefore FHE provides a measure of differential privacy for free. An earlier version of this review advanced a version of that argument. It is wrong, and because the error is seductive and consequential for system design, this subsection states plainly why, and then identifies what the correct composition of the two mechanisms actually looks like.

4.3.1 *Why the analogy fails*

Table 8 separates the two guarantees along the axes that a systems engineer would otherwise conflate.

Table 8. FHE and differential privacy defend against disjoint adversaries. The two mechanisms are complementary, not substitutable, and the earlier literature that treats ciphertext error as an implicit privacy budget conflates the two columns

	FHE	Differential privacy
Protects against	The party computing on the data	The party receiving the result
Adversary class	Polynomial-time, without sk	Unbounded, including the key holder
Guarantee type	Computational, under LWE	Information-theoretic
Noise calibrated to	Security level and noise budget	Query sensitivity Δf and ε
Noise located in	Ciphertext ring R_q; removed by decryption	Query output; persists after release
Survives decryption	No	Yes
Requires clipping	No	Yes
Composes over queries	Not applicable	Yes, via privacy accounting

In BGV and BFV a freshly encrypted value m is encoded as $\boldsymbol{c} = ([\boldsymbol{as} + \boldsymbol{e} + \Delta m]_q,\ [\boldsymbol{a}]_q)$ with $\boldsymbol{e} \leftarrow \chi_\sigma^n$ a discrete Gaussian of standard deviation σ. Differential privacy [45] requires that for neighbouring datasets D, D' a mechanism $\mathcal{M}$ satisfy $\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] + \delta$. Both involve Gaussian noise. There the resemblance ends, and four independent obstructions block the inference.

4.3.2 *The noise is not calibrated to sensitivity*

The Gaussian mechanism sets $\sigma_{\mathrm{DP}} = c \times \Delta f / \varepsilon$, where Δf is the ℓ_2-sensitivity of the query, that is, the maximum change in output induced by altering one record. FHE error is chosen for an entirely unrelated reason: it is the smallest value that makes the LWE instance hard at the target security level while leaving usable noise budget. It has no functional dependence on Δf, on ε, or on any property of the dataset. Two datasets differing in one record produce ciphertexts whose error distributions are identical, because the error is sampled independently of the message.

4.3.3 *The noise is in the wrong space*

Differential Privacy (DP) noise is added to the output of a query, in the space where the analyst reads the answer. FHE error lives in the ciphertext ring R_q and is removed exactly by decryption. It is not a perturbation of the gradient; it is a perturbation of the encoding of the gradient, and correct decryption returns the gradient unperturbed to within the scheme's approximation error. A quantity that vanishes on decryption cannot bound the privacy loss of the decrypted value.

4.3.4 *No clipping, hence no bounded sensitivity*

Differential Privacy Stochastic Gradient Descent (DP-SGD) [37] obtains its guarantee only because per-example gradients are clipped to a fixed ℓ_2 norm before noise is added, which is what makes Δf finite and known. Homomorphic evaluation performs no clipping. Without a sensitivity bound there is no ε to compute, and the DP accounting machinery cannot be started at all.

4.3.5 *The guarantees quantify over different adversaries*

This is the distinction Reviewer comment on computational versus information-theoretic security turns on, and it is worth stating carefully. FHE security is computational: it holds against probabilistic polynomialtime adversaries under the LWE assumption, and it holds only for adversaries who do not possess sk. DP is information-theoretic: it holds against computationally unbounded adversaries, it requires no hardness assumption, and it continues to hold after the output is released in the clear. The two protect against disjoint threats. FHE protects the data from the party performing the computation. DP protects the data from the party receiving the result. Neither substitutes for the other, and an architecture that needs both must deploy both.

Let Π be an IND-CPA secure FHE scheme and let ∇_D denote the gradient computed on dataset D. For any two datasets D, D' and any probabilistic polynomial-time adversary $\mathcal{A}$ that does not hold sk,

$$|\Pr[\mathcal{A}(\mathrm{Enc}(\mathrm{pk},\ \nabla_D)) = 1] - \Pr[\mathcal{A}(\mathrm{Enc}(\mathrm{pk},\ \nabla_{D'})) = 1]| \leq \mathrm{negl}(\lambda).$$

Proposition 4.1 is a restatement of semantic security and nothing more. It is stated explicitly here so that it cannot be mistaken for a privacy result. It says that an adversary without the key learns nothing from the ciphertext, which is true of the encryption of any value whatsoever, including one computed with no privacy protection at all. It establishes no membership privacy: the moment the model is decrypted by the legitimate key holder, the guarantee is discharged in full and the plaintext model is exactly as vulnerable to membership inference [34] and extraction [32] as a model trained with no encryption.

4.3.6 *Correct composition*

The engineering consequence is a strict ordering. Differential privacy must be applied to the plaintext gradients, before encryption, using clipping and calibrated noise in the usual way; encryption then protects those already private gradients from the compute provider. Yu et al. [46] establish that this is practical at language-model scale, and their finding is directly relevant to the protocol of Section 4.3: restricting the trainable parameters to a low-rank adaptation both improves the privacy-utility tradeoff and reduces the dimensionality over which the privacy budget must be spent, bringing private models close to non-private utility on standard tasks. The parameter-efficiency that motivates LoRA under encryption is therefore the same property that makes DP tractable, which is a genuine alignment between the two mechanisms and the strongest available argument for the combined architecture. Formally, if $\mathcal{M}_{\mathrm{DP}}$ is (ε, δ)-DP then $\mathrm{Enc}(\mathrm{pk},\ \mathcal{M}_{\mathrm{DP}}(D))$ retains the $(\varepsilon,\ \delta)$ guarantee after decryption, because post-processing cannot degrade DP. The reverse order provides nothing: encrypting first and hoping the ciphertext error suffices yields no ε at any δ.

This ordering carries a real cost that the free-privacy framing obscures. Clipping and noise addition are nonlinear operations on the plaintext, so under encrypted training they must either be performed client-side before encryption, which constrains the protocol to settings where a trusted client holds the data, or evaluated homomorphically, which requires a comparison circuit for the clipping norm and is among the more expensive operations in the CKKS setting. Section 4.3.2 lists this among the unresolved circuit requirements.

A separate line of work does establish DP guarantees from discrete Gaussian noise [47], and it is worth noting why it does not rescue the analogy. Those results calibrate a discrete Gaussian to a sensitivity bound and prove a hockey-stick divergence bound for that calibrated distribution. The mechanism is deliberately constructed to be private. FHE error is not constructed that way, and no choice of LWE parameters makes it so, because the parameter that would have to depend on Δf is fixed by the security requirement instead.

4.3.7 *Membership inference under encrypted training*

Membership inference attacks target plaintext model outputs. During encrypted inference the server returns $\mathrm{Enc}(\mathrm{pk},\ f_\theta(x))$ rather than $f_\theta(x)$, so a server that does not hold sk cannot observe confidence scores and cannot mount the attack. This is a genuine and useful property, and it is the correct claim to make for FHE. It is a statement about the server, not about the model.

Any party holding sk, or with access to a decryption oracle, mounts membership inference exactly as against an unencrypted model. Carlini et al. [32] established that training sequences can be extracted verbatim, and that within the model family they studied the larger models memorized more and complete memorization occurred after as few as thirty-three insertions of a sequence; the later systematic study of Carlini et al. [33] establishes log-linear growth of memorization in model capacity, in the number of times an example is duplicated, and in the length of the prompting context. Nothing in the encrypted training pipeline alters those relationships, because the decrypted model is a function of the same gradients. This review makes no claim that homomorphic evaluation reduces memorization. No experiment reported here or, to the authors' knowledge, in the published literature demonstrates such an effect, and the mechanism by which it could arise is unclear given that decryption is designed to recover the gradient exactly.

4.4 *Computational overhead analysis*

Table 9 summarizes the overhead of FHE operations relative to plaintext equivalents, based on published benchmarks for CKKS on commodity hardware. These figures represent single-threaded CPU estimates. For a $d = 768$ matrix-vector product, the standard diagonal packing method requires $\lceil\sqrt{768}\rceil \approx 28$ homomorphic rotations; at $N = 2^{15}$ each rotation costs roughly 15-30 ms on a single CPU thread (measured on OpenFHE/SEAL), placing the rotation budget alone at 420-840 ms. GPU acceleration (e.g., cuFHE, OpenFHE on GPU) reduces these numbers by one to two orders of magnitude. Larger reductions have been reported for dedicated silicon: Cheetah [48] obtains up to 79× over GAZELLE [49] from parameter tuning and operator scheduling alone, and projects five to six orders of magnitude for its proposed ASIC. Those figures are for convolutional networks rather than transformers, and the ASIC is a design study rather than fabricated hardware.

Table 9. Approximate FHE overhead for key LLM operations (CKKS, $N = 2^{15}$, 128-bit security, single-threaded CPU baseline; GPU/FPGA acceleration reduces slowdown by $10\times - 100\times$). Matrix-vector timing reflects the diagonal-method rotation budget of ≈ 28 rotations at ∼15-30 ms each

Operation	Plaintext time	FHE time	Slowdown
Matrix-vector mult ($d = 768$)	< 0.1 ms	500 ms	$\sim 5 \times 10^3\times$
GELU (degree-7 poly approx.)	< 0.1 ms	~ 500 ms	$\sim 5 \times 10^3\times$
Bootstrapping	N/A	2,000 ms	-
LoRA rank-8 update	< 0.5 ms	800 ms	$\sim 1.6 \times 10^3\times$

Figure 6 traces the corresponding noise budget, which reaches zero after roughly eight multiplications and forces a bootstrapping step. These overheads are prohibitive for large-scale training but are being reduced by hardware accelerators [48] and by algorithmic work on the non-polynomial operations that dominate the cost. PEGASUS [50] is representative of the latter: it switches between packed CKKS ciphertexts and FHEW ciphertexts without decryption, so that polynomial and non-polynomial parts of a computation are each evaluated under the scheme that suits them. Its reported applications are decision-tree evaluation and K-means rather than transformers, so its relevance here is as a technique rather than as a demonstrated result on attention. The trajectory suggests that FHE-based private fine-tuning of small LLMs (fewer than 1 B parameters) with LoRA may become practical within a few years.

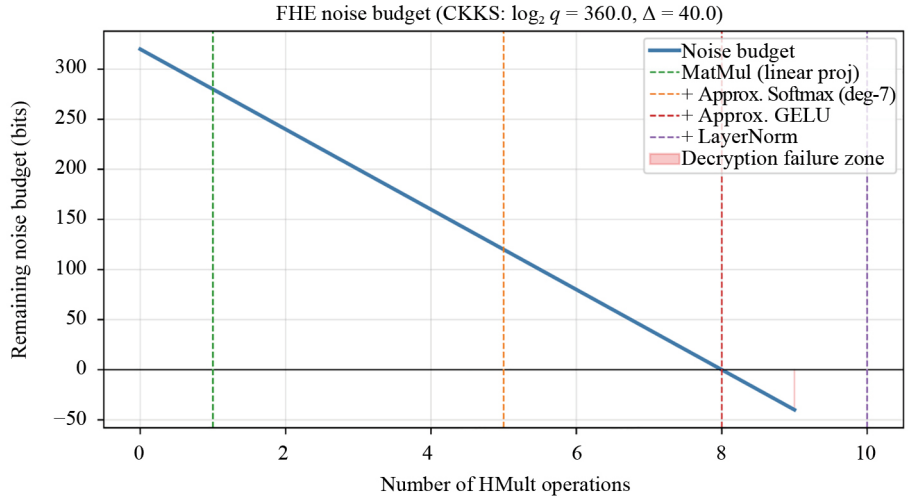

Figure 6. CKKS noise budget consumption across homomorphic multiplications for a single ciphertext chain $(\log_2 q = 360,\ \Delta = 2^{40})$. Each transformer operation consumes a fixed budget slice; the budget reaches zero after approximately eight multiplications, requiring bootstrapping to continue

4.5 *State of the art in encrypted transformer inference*

Encrypted inference has advanced substantially since the constructions of Section 4.2 were introduced, and the gap between it and the encrypted training proposal of Section 4.3 is now the clearest way to see why the latter remains distant.

BOLT [51] combines homomorphic encryption with secret sharing for transformer inference, reporting a 10.9× reduction in communication and 4.8 to 9.5× faster inference than prior systems. It is interactive, so its cost profile is bound by network round trips in the manner described in Section 7.

NEXUS [40] removes the interaction entirely, requiring the client only to submit an encrypted input and await an encrypted result. It contributes an argmax algorithm reducing complexity from $O(m)$ to $O(\log m)$ in the label count, which matters because $m = 30,522$ for BERT-base, and reports a GPU-accelerated per-token cost of a few cents.

THOR [39] is non-interactive and addresses precisely the non-linear evaluation problem identified in Section 4.3.2, giving strategies for softmax, LayerNorm, GELU, and Tanh through approximation combined with adaptive iterative methods. It reports secure inference on BERT-base at 128 tokens in ten minutes on a single GPU.

Two conclusions follow, and they cut in opposite directions. The optimistic one is that the non-linear circuits this review lists as unresolved are not uniformly unsolved: THOR demonstrates workable strategies for several of them at inference time, which is genuine progress and reduces the M2 milestone of Table 19 from open to partially addressed.

The sobering one concerns magnitude. Ten minutes for a single forward pass of a 110M-parameter encoder, on a GPU, is roughly five orders of magnitude above the plaintext cost: an unencrypted BERT-base forward pass at 128 tokens runs in single-digit milliseconds on comparable hardware, so 600 s against roughly 5 ms gives a ratio near 10^5. Fine-tuning requires a forward pass, a backward pass, and an update, repeated for thousands of steps. Applying that structure to the THOR figure places encrypted fine-tuning of even a small model in the range of years of GPU time at present efficiency.

This is the quantitative basis for the TRL 2 assignment in Table 18, and it indicates that the barrier is not a missing circuit but a throughput deficit that no incremental improvement closes.

4.6 *Engineering implementation profile*

Table 10 provides a Technology Readiness Level (TRL) assessment and hardware co-design profile for FHE-protected LLM training, structured for practitioners in computational engineering and secure systems design.

Table 10. Engineering implementation profile for FHE-protected LLM operations. TRL follows the NASA/DoD 1-9 scale: TRL 1-3 = basic research, TRL 4-6 = validation, TRL 7-9 = deployment-ready. These are component-level ratings. Where a component corresponds to one of the three frontiers, the frontier-level assignment in Table 18, made against the stricter criteria of Table 17, governs

Component	Primary engineering challenge	TRL	Recommended accelerator
CKKS bootstrapping	Noise-budget reset, circuit depth	4	FPGA (Xilinx U280), custom NTT unit
FHE linear layer ($d = 768$)	Number Theoretic Transform throughput	5	GPU (cuFHE), ASIC prototype
FHE LoRA adapter	Rank compression, key reuse	2	GPU + AVX-512 CPU
Poly. approx. (GELU/softmax)	Degree-accuracy tradeoff	4	CPU (OpenFHE scalar backend)
Multi-key FHE training	Key aggregation, bandwidth	3	Distributed GPU cluster

For a systems engineering budget, the dominant cost driver is the Number Theoretic Transform (NTT) at the core of every CKKS multiplication. An NTT on a dimension-N ciphertext ($N = 2^{15}$, 128-bit security) requires $O(N \log N)$ modular multiplications and dominates FPGA and GPU roofline analyses. The engineering path to practical deployment runs through three milestones: (i) NTT hardware specialization reducing the matrix-vector multiplication overhead from $10^3\times$ to $10\times$; (ii) fused bootstrapping pipelines that amortize the 2-second refresh cost across minibatches; and (iii) structured parameter formats (LoRA, prompt tuning) that limit the encrypted circuit depth to fewer than ten multiplications per gradient step.

Current open-source stacks (OpenFHE, SEAL, Lattigo) have reached TRL 5 for non-LLM workloads; adaptation to transformer fine-tuning remains at TRL 2, for the throughput reason set out above.

4.7 *Open challenges*

1. Non-linear function approximation. Polynomial approximations of softmax and GELU introduce approximation error that accumulates across layers. High-degree Chebyshev approximations are more accurate but consume more noise budget. Probabilistic approximation methods [38] offer improved accuracy-depth tradeoffs for functions such as Sign and Compare.

2. Backpropagation under FHE. Computing gradients requires evaluating the backward pass, which roughly doubles the circuit depth and noise budget requirement compared to the forward pass alone.

3. Key management in multi-party training. When training data is held by multiple parties, a threshold FHE or multi-key FHE protocol [52] is required so that no single party holds the master secret key.

4. Post-decryption privacy. FHE protects data in transit and during computation. Guarantees after decryption require composing FHE with formal DP, which has not been rigorously formalized for the LLM fine-tuning setting.

5. Verification of FHE evaluation. A malicious compute server may return an incorrect ciphertext. Verifiable FHE [53] addresses this but adds further overhead.

5. LLM-guided heuristics for lattice cryptanalysis

The second frontier reverses the usual role of LLMs in AI security: rather than protecting LLMs, we ask whether LLMs can be used to attack the cryptographic assumptions that protect everything else. Concretely, the question is whether

an LLM, acting as an adaptive black-box optimizer, can make lattice reduction algorithms such as BKZ more efficient by choosing better reduction strategies online. Even a modest improvement in BKZ efficiency would narrow the security margins of NIST-standardized lattice-based schemes and would require practitioners to increase LWE parameters, raising computational costs for FHE and FE applications.

5.1 *Problem formulation*

Post-quantum security claims depend critically on the concrete hardness of LWE and related lattice problems at specific parameter sizes. The security level λ (in bits) is estimated by computing the smallest BKZ block size β^* that would solve the corresponding lattice problem within time $T \approx 2^{c\beta^*}$. This estimation is inherently heuristic: both the Gaussian heuristic for lattice point counts and the BKZ simulator involve approximations that may underestimate the power of future algorithms.

The open question this section addresses is: can an LLM, acting as an adaptive search heuristic, identify better reduction strategies for a given lattice instance than those achievable by purely algorithmic lattice reduction?

5.2 *The BKZ algorithm and its limitations*

Before describing how LLMs can be incorporated, we review the BKZ algorithm and identify the three heuristic choices that dominate its practical performance. Each of these choices represents a degree of freedom that a learning-based strategy could potentially exploit. Algorithm 2 describes BKZ at a high level. The key parameters are the block size β and the number of tours (full passes over all blocks).

Algorithm 2:
Require: Basis $\boldsymbol{B} \in \mathbb{Z}^{n \times n}$, block size β
Ensure: Reduced basis $\boldsymbol{B}'$ with $\left\|\boldsymbol{b}'_1\right\| \leq \gamma_\beta^{n-1} \times \det(\Lambda)^{1/n}$
$\boldsymbol{B}' \leftarrow \mathrm{LLL}(\boldsymbol{B})$ ▷ Initial LLL reduction
for $t = 1$ to T **do**
  for $k = 0$ to $n - 1$ **do**
    $\pi_k \leftarrow$ orthogonal projection onto span $(\boldsymbol{B}'_1, \ldots, \boldsymbol{B}'_k)^{\perp}$
    $\mathbf{v} \leftarrow \mathrm{SVP}(\pi_k(\Lambda_k))$ ▷ SVP oracle on block of dim. β
    Insert $\mathbf{v}$ into basis and reduce via LLL
  end for
end for
return $\boldsymbol{B}'$

Three heuristic choices determine BKZ performance in practice:

1. SVP oracle strategy: Enumeration or sieving within each block.

2. Block size schedule: Whether β is fixed or progressively increased (progressive BKZ [54]).

3. Preprocessing: The quality of the initial LLL basis affects all subsequent reductions.

Each of these choices is typically fixed by hand or determined by a limited grid search. The hypothesis motivating this frontier is that an LLM, presented with the runtime profile and Gram-Schmidt Orthogonalization (GSO) norms of the current basis, can predict better choices online.

5.3 *LLMs as black-box optimizers for lattice reduction*

We now describe a concrete framework for using an LLM as an online strategy selector for BKZ. The framework has three components: a compact state representation that encodes the current basis quality, a reinforcement learning formalism that turns strategy selection into a structured decision problem, and a prompting protocol that queries the LLM at each reduction step.

5.3.1 *Problem encoding*

The state of a BKZ reduction after each tour can be compactly described by the GSO log-norms $\log \|\boldsymbol{b}_i^*\|$ for $i = 1, \ldots, n$ (following BKZ literature notation; $\boldsymbol{b}_i^*$ is equivalent to $\widetilde{\boldsymbol{b}}_i$ from Section 3). A reduced basis approaches the Gaussian heuristic profile:

$$\log \|\boldsymbol{b}_i^*\| \approx \frac{n-1-2i}{2n} \operatorname{logdet}(\Lambda) + \operatorname{logvol}(\mathcal{B}_n)^{1/n}.$$

The deviation from this profile at each index is a rich feature vector that characterizes the current quality of the reduction and hints at which blocks are most productive to attack next.

5.3.2 *LLM as a reinforcement learning policy*

We frame BKZ strategy selection as a Markov Decision Process (MDP):

$$\text{State } s_t = (\log \|\boldsymbol{b}_1^*\|, \ldots, \log \|\boldsymbol{b}_n^*\|) \text{ after tour } t$$

$$\text{Action } a_t = (\beta_t, k_t, \text{ oracle}_t) \in \{32, \ldots, n\} \times \{0, \ldots, n-\beta_t\} \times \{\text{enum}, \text{ sieve}\}$$

$$\text{Reward } r_t = \|\boldsymbol{b}_1^*\|_{t-1} - \|\boldsymbol{b}_1^*\|_t.$$

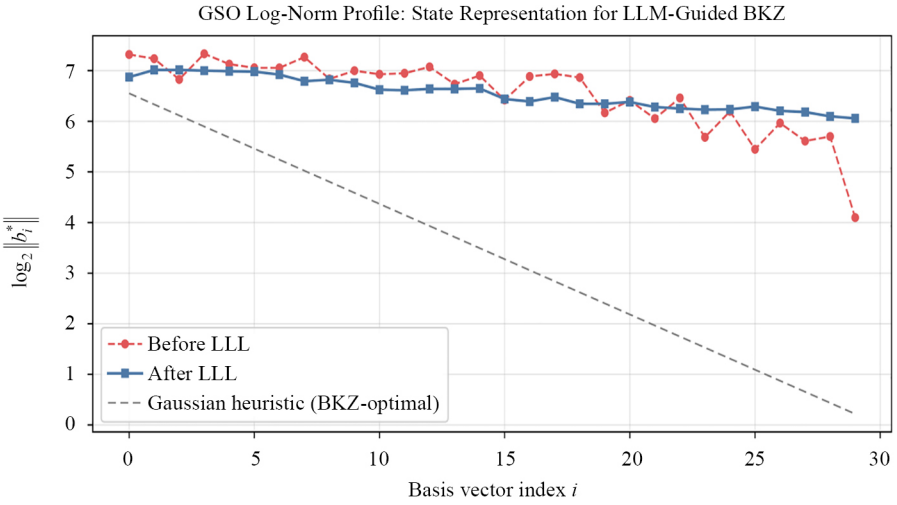

Figure 7. Gram-Schmidt log-norm profiles before and after LLL reduction on a 30-dimensional random lattice. The gap between the post-LLL profile and the Gaussian heuristic (BKZ-optimal) slope represents the opportunity for LLM-guided BKZ to selectively apply larger block sizes at high-deviation indices

Figure 7 shows the Gram-Schmidt profile that motivates this encoding: the deviation from the Gaussian heuristic is uneven across indices, which is what an adaptive schedule would exploit. An LLM is prompted with a serialized representation of s_t and asked to output a_{t+1}. The structure of this loop is conceptually analogous to the FunSearch approach [55], where an LLM proposes successive modifications to candidate programs that are evaluated by an external scoring mechanism. This analogy does not imply that LLMs have demonstrated improvements to lattice reduction

algorithms. The analogy is offered as a design template only. FunSearch was applied to the cap set and online bin packing problems; it has not been applied to lattice reduction or to cryptographic parameter optimization, and nothing in that work should be read as evidence that the transfer succeeds.

Algorithm 3:
Require: Lattice basis $\boldsymbol{B}$, LLM policy π_{LLM}, budget T
Ensure: Short basis vector $\boldsymbol{b}'_1$
$\boldsymbol{B}' \leftarrow \mathrm{LLL}(\boldsymbol{B})$
for $t = 1$ to T **do**
  $s_t \leftarrow \left(\log \|\boldsymbol{b}'^{*}_i\|\right)_{i=1}^{n}$
  $(\beta_t, k_t, \mathrm{oracle}_t) \leftarrow \pi_{\mathrm{LLM}}(s_t, \mathrm{history})$ ▷ LLM selects action
  $\boldsymbol{B}' \leftarrow \mathrm{BKZ}_{\beta_t}(\boldsymbol{B}', \mathrm{start} = k_t, \mathrm{oracle} = \mathrm{oracle}_t, \mathrm{tours} = 1)$
end for
return $\boldsymbol{B}'[0]$

5.3.3 *Prompting strategy*

Figure 8 shows the resulting interaction loop, and Algorithm 3 states it in full. The LLM prompt for each step consists of three components:

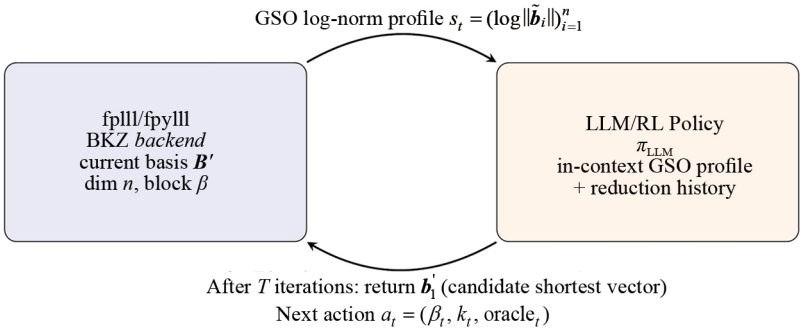

Figure 8. LLM-guided BKZ interaction loop. At each reduction step, the fpllll backend serializes the current Gram-Schmidt log-norm profile s_t and sends it to the LLM policy π_{LLM}, which returns the next block size β_t, starting index k_t, and SVP oracle type (enumeration or sieving). This replaces the rigid, pre-computed block-size schedule of standard BKZ with an adaptive, context-sensitive strategy

1. System context: A description of BKZ, the GSO profile structure, and what a good reduction looks like.

2. State serialization: The current GSO log-norm vector as a JSON array with statistical summary statistics (min, max, monotonicity violations).

3. Action request: "Given this profile, suggest (block_size, start_index, oracle) for the next BKZ step to minimize $\log\|b_1^*\|$."

5.4 *Implications for post-quantum security estimation*

The concrete security of NIST-standardized schemes is estimated using the LWE estimator tool [15] together with a BKZ cost model [56]. It is tempting to reason that a scheduler which reaches a given reduction quality in fewer tours thereby weakens the scheme. That reasoning is invalid, and since an earlier version of this review relied on it, the error is worth setting out in full.

5.4.1 *Why fewer tours does not mean lower security*

Concrete security under the core-SVP methodology is governed by β^*, the smallest block size at which the attack succeeds at all. The dominant cost is the single SVP oracle call on a block of that size, at $2^{0.292\beta^*}$ for sieving. The number of tours enters the cost only as a polynomial factor, and it does not enter the determination of β^* at all.

The value of β^* is fixed by lattice geometry. Under the geometric series assumption, a BKZ-β reduced basis has a root Hermite factor $\delta_\beta \approx \left(2/\pi e(\pi\beta)^{1/\beta}\right)^{1/(2(\beta-1))}$, and the attack succeeds when δ_β is small enough to expose the secret against the target lattice. This is a statement about the fixed point that BKZ-β converges to, not about the trajectory taken to reach it. A scheduler that arrives at the same fixed point in half the tours has saved a constant factor of wall-clock time and has changed δ_β, and therefore β^*, not at all.

The consequence is stated as a proposition because it bounds what this entire frontier could achieve even in the best case.

Let π be any block-size and index schedule whose SVP oracle calls all use block size at most β. Then the basis quality attainable by π is bounded by that of full BKZ-β reduction, and the core-SVP cost estimate $2^{0.292\beta}$ is unchanged. Any speedup from π is a reduction in the number of oracle calls, which affects the polynomial factor only.

A scheduling policy could therefore only affect concrete security by making a smaller block size suffice where a larger one was previously required, which is a claim about the achievable root Hermite factor and not about tour count. No result in the literature demonstrates such an improvement, and the mechanism by which adaptive scheduling could produce one is not apparent, since the limiting quantity is a property of the oracle rather than of its scheduling.

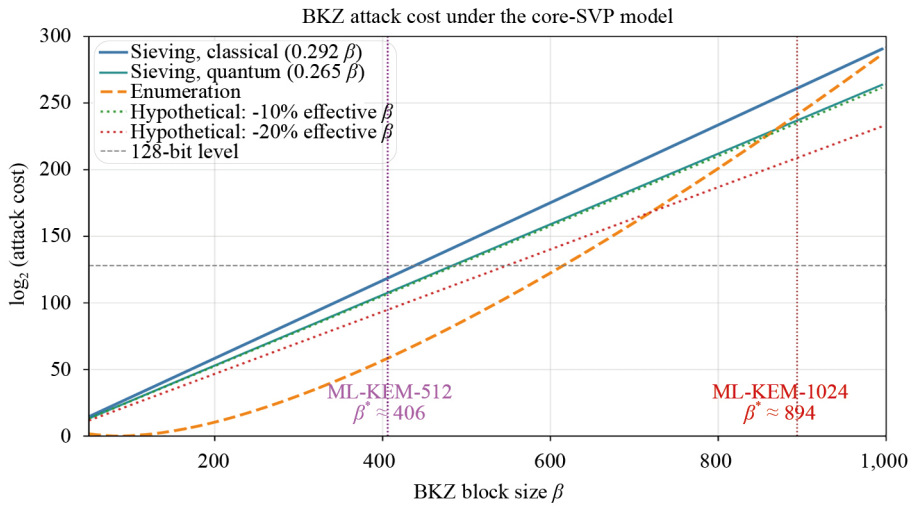

Figure 9. BKZ attack cost in $\log_2$ operations against block size β under sieving and enumeration models. The curves labelled as reduced-β scenarios are hypothetical illustrations only: they show what the cost would be if some future technique made a smaller block size sufficient, and they are not predictions, measurements, or projections from any observed LLM behaviour. No LLM-guided reduction experiment underlies this figure. By Proposition 5.1 an adaptive scheduler alone cannot move a system along these curves

Figure 9 plots attack cost against block size under the core-SVP convention, and makes the point geometrically: the curves are indexed by β, so a scheduler that changes only the order of operations does not move a system along them.

Breaking ML-KEM-512 requires a block size of $\beta^* \approx 400$ or above, giving sieving complexity of order $2^{0.292\beta^*}$. This review reports no evidence that LLM-guided reduction lowers β^* by any amount, and Proposition 5.1 indicates that scheduling improvements alone cannot. The correct engineering posture is therefore that this frontier presents *no* current basis for adjusting parameter selection. Deployed systems should follow the LWE estimator and NIST category guidance without modification on account of this work.

5.5 *Related work: ML for lattice problems*

Several prior works have applied neural networks to lattice problems without using LLMs.

5.5.1 *Nearest-neighbour structure in sieving*

The sieving algorithms that determine the cost of the SVP oracle rely on structured nearest-neighbour search techniques: Becker et al. [57] derive the $2^{0.292\beta}$ heuristic running-time estimate by reducing lattice sieving to nearest-neighbour search with locality-sensitive filtering over the sphere. The filters are combinatorially designed rather than learned, and replacing that design with a learned analogue is the most concrete point at which machine learning could in principle affect the constant in the exponent. No such result exists, and because that constant is what Proposition 5.1 identifies as the binding quantity, this rather than schedule selection is where a genuine cryptanalytic advance would have to come from.

5.5.2 *Learning-based preprocessing*

The idea of using a learned model to select the preprocessing basis before BKZ was explored with classical ML (random forests, gradient-boosted trees) in the context of Boolean Satisfiability Problem (SAT) and related combinatorial problems, but has not been systematically applied to lattice reduction.

5.5.3 *NeuroSAT and combinatorial optimization*

The established result is that graph neural networks can be trained to predict satisfiability of SAT instances from single-bit supervision, and that the learned representations recover satisfying assignments [58]; comparable architectures have been applied to other combinatorial problems. The remainder of this paragraph is a proposal of the present authors and is not a finding of that work: a lattice basis can be embedded as a weighted graph, with nodes as basis vectors and edge weights as inner products, which would give a Graph Neural Network (GNN) a natural input format. No such model has been trained or evaluated here or, to the authors' knowledge, elsewhere.

5.5.4 *LLM-based mathematical search*

The FunSearch system [55] demonstrated that an LLM placed in an evaluator-guided loop can discover new constructions in extremal combinatorics, improving the best known lower bound on the cap set problem and finding heuristics for online bin packing. Whether that paradigm transfers to the BKZ parameter schedule is the open question this section poses; the structural similarity is suggestive, and it is not evidence.

5.5.5 *A falsifiable benchmark protocol*

The hypothesis of this section is unsupported, and the appropriate response is not to hedge it but to specify what would settle it. This subsection states a benchmark protocol precise enough that a negative result would be as publishable as a positive one. It is offered as the concrete next step for anyone wishing to test the paradigm, and its inclusion is the substantive contribution of Frontier 2 in the absence of experimental results.

5.5.5.1 *Null hypothesis*

H_0: for a fixed SVP oracle and a fixed total oracle-call budget, an LLM-driven block-size and index schedule attains a root Hermite factor no better than the best static BKZ-β schedule under the same budget. Rejecting H_0 requires a strictly smaller δ at equal budget, at a significance level fixed in advance.

5.5.5.2 *Lattice instances*

Three families, since transfer across geometry is itself in question: uniformly random q-ary lattices; LWE instances from the ML-KEM parameter family scaled down in dimension; and NTRU lattices. Dimensions $n \in \{60, 80, 100, 120, 140\}$, which span the range where exhaustive comparison against optimal static schedules remains feasible. Thirty independent instances per dimension per family, seeds published.

5.5.5.3 *Baselines*

Four, and the third is the one that matters. (i) LLL alone. (ii) Static BKZ-β swept over β with the best result reported, which is the fair comparison rather than an arbitrary fixed β. (iii) BKZ 2.0 with progressive block sizes and its standard preprocessing and pruning [27]. (iv) A random schedule drawn from the same action space as the LLM policy, which controls for the possibility that any adaptivity helps regardless of the policy's content.

5.5.5.4 *Budget accounting*

Comparison must be at equal cost, and cost must be counted in SVP oracle calls weighted by $2^{0.292\beta}$ rather than in wall-clock time or tour count. Reporting wall-clock time invites the tours fallacy of Section 5.4.1; reporting unweighted call counts lets a policy appear efficient by making a few very expensive calls. LLM inference cost is reported separately and is not charged against the lattice budget, which biases in favour of H_1 and is stated so that readers can discount accordingly.

5.5.5.5 *Primary metric*

Root Hermite factor δ achieved at fixed weighted budget, aggregated across seeds with confidence intervals. Secondary metrics: the smallest β at which the secret is recovered, and the shape of the resulting GSO profile against the geometric series assumption prediction.

5.5.5.6 *Falsification criterion*

The hypothesis is refuted if, across all three lattice families, the LLM policy fails to beat baseline (ii) at any tested dimension, or if any advantage observed at small n fails to persist as n grows. The latter is the more likely failure mode and the more important to report: a technique that helps at $n = 60$ and not at $n = 140$ is of no cryptanalytic interest, since the dimensions that matter exceed 500.

5.5.5.7 *Reporting requirements*

Model identity and version, full prompts, temperature, seeds, the fplll or G6K version used for the oracle, hardware configuration, and the complete action log for every run. Without these, a reported improvement cannot be distinguished from a favourable draw.

Applying this protocol is beyond the scope of a review article, and this review does not claim to have applied it. Until someone does, the statements in Section 5.3 remain in evidence class E4 and carry no implications for parameter selection.

5.6 *Open challenges*

1. State space size. The GSO profile has n real-valued entries. For cryptographically relevant dimensions ($n = 512$ to $2,048$), this is a high-dimensional input that may exceed LLM reasoning capacity without further compression.

2. Oracle call budget. Each SVP oracle call on a block of size β costs $2^{O(\beta)}$ time. An LLM that proposes unnecessarily large β wastes budget; one that proposes β too small makes no progress. Calibrating the LLM's cost awareness is a key open problem.

3. Adversarial implications. If LLM-guided lattice reduction becomes significantly more powerful, it could invalidate current security proofs. The community needs principled methods to bound any improvement that a learning-based strategy can achieve over optimal non-adaptive BKZ.

4. Transfer across lattice families. LWE lattices, NTRU lattices, and ideal lattices have different geometric structure. A strategy tuned on one family may not transfer, requiring separate training or prompting per scheme.

5. Verifiable reduction quality. Unlike pure algorithmic BKZ, an LLM-guided approach does not guarantee monotone progress. A certificate of reduction quality (e.g., comparing against the LLL bound) is needed to bound failure probability.

6. Lattice-based functional encryption for decentralized LLM inference

The third frontier addresses a structural limitation of the first two: FHE requires an all-or-nothing trust model (the server either sees everything or computes entirely on ciphertexts), while LLM-guided attacks expose a security risk concentrated in one algorithm. Functional encryption provides a more granular alternative. In a split-inference pipeline with K compute nodes, each node should be able to evaluate exactly its assigned transformer block and nothing more. Lattice-based inner-product FE makes this possible with provable security under the LWE assumption.

6.1 *Problem formulation*

In a decentralized LLM inference setting, the computation of $\hat{y} = f_\theta(x)$ is distributed across K untrusted nodes $\mathcal{P}_1, \ldots, \mathcal{P}_K$, each responsible for a contiguous block of transformer layers $\ell \in [l_{k-1}+1,\ l_k]$. The security requirements are:

1. Input privacy: Node $\mathcal{P}_k$ learns only the intermediate activation tensor entering its block, not the original prompt x.

2. Weight privacy: Node $\mathcal{P}_k$ should not learn the full weight matrices of other nodes (or, in the strongest setting, its own).

3. Output correctness: The final output $\hat{y}$ is identical to that of centralized inference, up to rounding.

Functional encryption provides an elegant solution: the model owner issues a functional decryption key sk_{f_k} to node $\mathcal{P}_k$ that enables computing exactly $f_k(\boldsymbol{h}_{l_{k-1}}) = \boldsymbol{h}_{l_k}$, the k-th block's forward pass, without decrypting the full hidden state.

6.2 *Inner-product functional encryption for linear layers*

The linear projection layers of a transformer are the natural fit for inner-product FE, since each output neuron is a dot product between a weight vector and the input activation. The ALS scheme [25] provides an LWE-based construction with adaptive simulation security, whose algebraic structure we describe next. The building block is Inner-Product FE (IPFE) [25]. In a transformer, the linear projection layers compute:

$$\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x} = \begin{pmatrix} \langle \boldsymbol{w}_1,\ \boldsymbol{x} \rangle \\ \vdots \\ \langle \boldsymbol{w}_d,\ \boldsymbol{x} \rangle \end{pmatrix},$$

where $\boldsymbol{w}_i$ is the i-th row of $\boldsymbol{W}$. An IPFE scheme allows a node holding a functional key $\mathrm{sk}_{\boldsymbol{y}} = \mathrm{FKGen}(\mathrm{msk},\ \boldsymbol{y})$ to compute $\langle \boldsymbol{y},\ \boldsymbol{x} \rangle$ from $\mathrm{Enc}(\mathrm{pk},\ \boldsymbol{x})$ alone.

6.2.1 *Lattice-based IPFE construction*

The Agrawal-Libert-Stéhlé IPFE scheme [25] is based on the LWE assumption and Achieves Adaptive Simulation Security (AD-SIM):

Let n be the LWE dimension, m the number of samples, and q, B the modulus and message bound. Let χ_σ be a Gaussian error distribution and τ a discrete Gaussian over $\mathbb{Z}$. Following the parameter conditions of $m \geq 4n\log_2 q$. The scheme proceeds as follows:

$$\text{KeyGen}(\lambda):\ \boldsymbol{A} \leftarrow_q^{m\times n},\ \boldsymbol{Z} \leftarrow \tau^{\ell\times m},\ \boldsymbol{U} = \boldsymbol{Z}\boldsymbol{A} \in_q^{\ell\times n}$$

$$\text{Enc}(\text{pk},\ \boldsymbol{x}):\ \boldsymbol{s} \leftarrow_q^{n},\ \boldsymbol{e}_0 \leftarrow \chi_\sigma^m,\ \boldsymbol{e}_1 \leftarrow \chi_\sigma^\ell$$

$$:\ \boldsymbol{c}_0 = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}_0 \in_q^m,\ \boldsymbol{c}_1 = \boldsymbol{U}\boldsymbol{s} + \boldsymbol{e}_1 + \lfloor q/B \rfloor \times \boldsymbol{x} \in_q^\ell$$

$$\text{FKGen}(\text{msk},\ \boldsymbol{y}):\ sk_{\boldsymbol{y}} = \boldsymbol{Z}^\top \boldsymbol{y} \in {}^m$$

$$\text{Dec}(\text{sk}_{\boldsymbol{y}},\ \boldsymbol{c}):\ v = \boldsymbol{c}_1^\top \boldsymbol{y} - \boldsymbol{c}_0^\top sk_{\boldsymbol{y}} = \lfloor q/B \rfloor \langle \boldsymbol{x},\ \boldsymbol{y} \rangle + \text{small error}.$$

Rounding v to the nearest multiple of $\lfloor q/B \rfloor$ recovers $\langle \boldsymbol{x},\ \boldsymbol{y} \rangle$ when the error is smaller than $|q/(2B)|$. Note that $\boldsymbol{Z}$ is drawn from a discrete Gaussian over the integers rather than uniformly over $\mathbb{Z}_q$; the norm bound on $\text{sk}_{\boldsymbol{y}} = \boldsymbol{Z}^\top \boldsymbol{y}$ is what keeps the decryption error small, and a uniform $\boldsymbol{Z}$ would break correctness.

Decryption uses $\text{sk}_{\boldsymbol{y}}$ only modulo q, so a key may be stored in $m\lceil \log_2 q \rceil$ bits even though its entries as integers are considerably larger.

6.2.2 *Applying IPFE to transformer layers*

Each linear layer weight row $\boldsymbol{w}_i$ is registered as a functional key $\text{sk}_{\boldsymbol{w}_i}$. Node $\mathcal{P}_k$ holds $\{\text{sk}_{\boldsymbol{w}_i}\}_{i=1}^{d_{\text{out}}}$ and receives $\text{Enc}(\text{pk},\ \boldsymbol{h}_{\text{in}})$. It computes all $\langle \boldsymbol{w}_i,\ \boldsymbol{h}_{\text{in}} \rangle$ in parallel and assembles $\boldsymbol{h}_{\text{out}}$. Each functional key reveals only the inner product with a specific weight row. It does not follow that the node cannot reconstruct the hidden state: if the rows $\{\boldsymbol{w}_i\}_{i=1}^{d_{\text{out}}}$ span $\mathbb{Z}_q^{d_{\text{in}}}$, which holds for any full-rank projection, the node recovers $\boldsymbol{h}_{\text{in}}$ by solving a linear system. See Section 6.2.3.

The complete split-layer forward pass protocol for a sequence of K nodes proceeds in three explicit steps:

1. Encryption (user side). The user encrypts the input prompt embedding $\boldsymbol{x}$ under the lattice-based FE public key: $\text{ct}_0 = \text{Enc}(\text{pk},\ \boldsymbol{x})$. Only the final output $\hat{y}$ is returned in plaintext; all intermediate states remain cryptographically bound throughout.

2. Layer-wise computation (compute nodes). Node $\mathcal{P}_k$ holds the functional keys $\{\text{sk}_{\boldsymbol{w}_i^{(k)}}\}$ for its assigned transformer block. It evaluates each output neuron as $\langle \boldsymbol{w}_i^{(k)},\ \boldsymbol{h}_{k-1} \rangle$ via IPFE decryption, assembles the plaintext activation $\boldsymbol{h}_k$, applies the non-linear activation locally (GELU or SwiGLU), and re-encrypts before forwarding: $\text{ct}_k = \text{Enc}(\text{pk}_{k+1}, \boldsymbol{h}_k)$. The re-encryption is a point of interactivity in the pipeline. Proxy re-encryption, introduced by Blaze et al. [59] in the pre-lattice setting, is the standard primitive for delegating such a transformation without exposing the plaintext; realising it for the lattice-based ciphertexts used here would require a construction not supplied by that work and not instantiated in this review.

3. Decryption (output node). The last node decrypts the logit vector and applies the softmax to produce the output distribution $P(\hat{y}|x)$. Under the LWE assumption, no single node can reconstruct $\boldsymbol{x}$ or the hidden states of other nodes from its functional keys alone, yielding fine-grained, verifiable input privacy across the entire inference pipeline.

IPFE and MCFE natively support only linear inner-product computations. Non-linear operations (GELU, SwiGLU, softmax, LayerNorm) cannot be evaluated homomorphically under the standard IPFE construction. As described in Step 2 above, a compute node must **decrypt the intermediate activation to plaintext**, apply the non-linear function locally, and re-encrypt before forwarding. This decryption step exposes the intermediate hidden state $\boldsymbol{h}_k$ to that node and breaks end-to-end FE guarantees for those operations. Practical architectures must therefore choose one of three remedies: (a)

Trusted compute for non-linear layers with appropriate contractual or TEE-based guarantees; (b) A *hybrid FE + FHE* approach that switches to FHE (e.g., CKKS with polynomial activation approximation) for non-linear layers; or (c) *Architectural redesign* that replaces non-linear activations with bounded-degree polynomials compatible with FE. The security guarantees in the proposition below apply only to the linear-projection layers.

Under the LWE assumption with parameters (n, m, q, χ_σ), node $\mathcal{P}_k$ holding $\{\mathrm{sk}_{\boldsymbol{w}_i}\}$ and $\mathrm{Enc}(\mathrm{pk}, \boldsymbol{h})$ cannot distinguish $\boldsymbol{h}$ from any $\boldsymbol{h}'$ such that $\boldsymbol{W}\boldsymbol{h} = \boldsymbol{W}\boldsymbol{h}'$ with advantage greater than $\mathrm{negl}(\lambda)$.

6.2.3 *The guarantee is vacuous at full rank*

Proposition 6.1 is correct and, read carelessly, gravely misleading. It states that $\boldsymbol{h}$ is hidden within its equivalence class under $\boldsymbol{W}$. It says nothing about how large that class is, and for the key-issuance pattern that a working inference pipeline actually requires, the class is a single point.

Let $\boldsymbol{h} \in \mathbb{Z}_q^d$ and let a node hold functional keys for $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ with $\boldsymbol{W} = [\boldsymbol{w}_1; \ldots; \boldsymbol{w}_k] \in \mathbb{Z}_q^{k \times d}$. If $\mathrm{rank}(\boldsymbol{W}) = d$, then the node recovers $\boldsymbol{h}$ exactly as $\boldsymbol{h} = \boldsymbol{W}^+(\boldsymbol{W}\boldsymbol{h})$, where $\boldsymbol{W}^+$ is any left inverse. The recovery is a linear solve. It requires no cryptanalysis, does not violate the IND-CPA security of the scheme, and runs in $O(d^3)$ field operations.

This is not an attack on the cryptography. It is a consequence of what was asked of the cryptography. The scheme delivers exactly the functions it was instructed to deliver, and if those functions jointly determine the input, then delivering them determines the input. A transformer projection matrix $\boldsymbol{W}_Q \in \mathbb{R}^{d \times d}$ with $d = d_{\mathrm{model}}$ is square and generically full rank, so a node issued keys for a complete projection reconstructs the hidden state in full. The privacy claim of the architecture, as originally stated, therefore did not hold in the configuration the architecture requires.

Three mitigations are available, and each costs something real.

6.2.4 *Rank-deficient key issuance*

Issue keys for at most $r < d$ independent functionals per node, so the residual uncertainty is a subspace of dimension $d - r$. This bounds leakage but limits each node to a rank-r projection, which for attention heads with $d_k = 64$ and $d_{\mathrm{model}} = 768$ is naturally satisfied by a single head but violated as soon as a node handles enough heads to span the space. The constraint is therefore an upper bound on how much work one node may be given, and it directly opposes the efficiency motivation for decentralization.

6.2.5 *Noise flooding*

Add calibrated noise to the released inner products so that the linear solve returns an approximate rather than exact $\boldsymbol{h}$. This is the only mitigation that degrades gracefully, and it is where functional encryption and differential privacy compose naturally, in contrast to the FHE case of Section 4.4: here the noise is on a released output, in the correct space, and can be calibrated to sensitivity. The cost is accuracy, and the accumulated perturbation across layers has not been characterized for transformer inference.

6.2.6 *Collision-aware allocation*

Since t colluding nodes pool their key sets, the rank constraint must hold for every coalition of size up to the tolerated threshold, not merely for each node individually. This is developed in Section 6.7.

The engineering conclusion is that the security of this architecture is governed by a combinatorial key-allocation constraint rather than by the LWE assumption. The lattice hardness is necessary and is not the binding constraint.

6.3 *Multi-client functional encryption for attention*

Single-ciphertext IPFE handles linear projections over a single encrypted vector, but self-attention requires computing pairwise interactions between all token positions simultaneously. Multi-client FE extends the single-client setting to allow joint computation over independently encrypted inputs from multiple parties, which matches the structure of a split-token attention mechanism.

The self-attention mechanism:

$$\mathrm{Attn}(\boldsymbol{Q},\, \boldsymbol{K},\, \boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\top}}{\sqrt{d_k}}\right)\boldsymbol{V},$$

requires cross-token interactions, which single-ciphertext IPFE does not naturally support. Multi-Client FE (MCFE) [60] allows an encryptor to encrypt multiple inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L$ (one per token position) under independent keys, and a decryptor to compute a function over all of them jointly.

6.3.1 *Why inner-product FE cannot realise attention*

A natural proposal is to have each shard i encrypt $\boldsymbol{h}_i$, issue the attention node functional keys for query-key interactions, and have it compute $\langle \boldsymbol{W}_Q \boldsymbol{h}_i,\, \boldsymbol{W}_K \boldsymbol{h}_j \rangle$ for all pairs $(i,\, j)$. This proposal does not work, and the reason is structural rather than a matter of efficiency.

Write the required quantity explicitly:

$$\langle \boldsymbol{W}_Q \boldsymbol{h}_i,\, \boldsymbol{W}_K \boldsymbol{h}_j \rangle = \boldsymbol{h}_i^{\top} \boldsymbol{W}_Q^{\top} \boldsymbol{W}_K \boldsymbol{h}_j = \boldsymbol{h}_i^{\top} \boldsymbol{M} \boldsymbol{h}_j, \quad \boldsymbol{M} = \boldsymbol{W}_Q^{\top} \boldsymbol{W}_K.$$

This is a bilinear form: degree one in $\boldsymbol{h}_i$ and degree one in $\boldsymbol{h}_j$, hence degree two overall in the plaintexts. Inner-product functional encryption, in both its single-client and multi-client forms, supports functionalities of the form $\boldsymbol{x} \mapsto \langle \boldsymbol{x},\, \boldsymbol{y} \rangle$ where $\boldsymbol{y}$ is fixed by the key. Such a functionality is linear in the plaintext. No sequence of linear functionals evaluates a quadratic form in the plaintexts, so Equation 21 lies outside the functionality class regardless of how many keys are issued or how they are combined.

Realising attention under functional encryption therefore requires quadratic or degree-2 functional encryption, a strictly stronger primitive. Lattice-based constructions for quadratic functionalities exist but are substantially less efficient than the inner-product case and lack the compact key structure that makes ALS attractive. This review does not claim a construction for encrypted attention, and the architecture of Section 6.4 should be understood as covering the linear projections of a transformer block only, with attention scores computed either in plaintext after decryption or by a different primitive entirely.

6.3.2 *Correction to the prior characterization*

An earlier version of this review described the DMCFE construction of Chotard et al. [60] as lattice-based and secure under RLWE. That is incorrect. That construction is built in pairing groups, drawing on private stream aggregation techniques, and its security rests on Diffie-Hellman-type assumptions in the random oracle model. It is consequently *not* post-quantum secure and does not belong to the lattice family that is the subject of this review. It is cited here as the canonical formulation of the decentralized multi-client inner-product functionality, not as a post-quantum instantiation of it. Constructing a lattice-based DMCFE with comparable decentralization properties remains open, and is listed among the open problems in Section 10.

6.4 *Decentralized inference architecture*

Figure 10 illustrates the full decentralized inference pipeline.

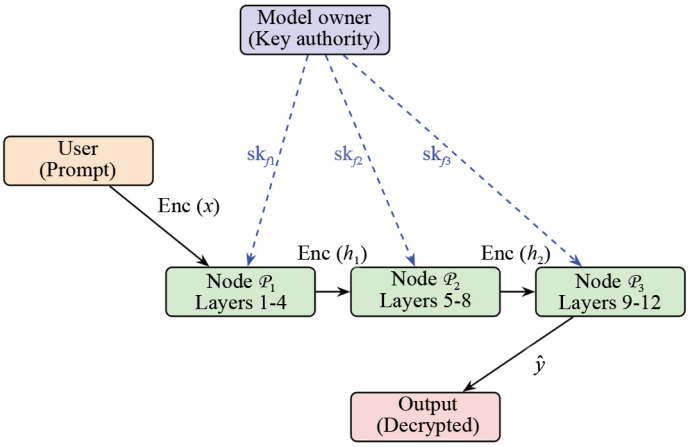

Figure 10. Lattice-based FE decentralized inference architecture. The model owner distributes functional keys to compute nodes, each of which evaluates one block of transformer layers. This is not an end-to-end encrypted pipeline: as detailed below, each node necessarily obtains plaintext activations at its own layer in order to apply non-linearities. The guarantee is that no node sees the original prompt and that each node's view is confined to its own layer, subject to the rank constraint of Proposition 6.2

6.4.1 *Key distribution*

The model owner acts as the trusted key authority. At deployment time, it issues functional decryption keys to each compute node. Key updates, for instance after fine-tuning, require re-issuing affected keys, which is efficient for LoRA adapters since only the low-rank corrections change. Issuance is subject to the rank bound of Section 6.2.3, which must hold per coalition rather than per node.

6.4.2 *Forward pass and the decryption boundary*

Each node decrypts its layer outputs via FE, obtaining plaintext activations, applies its non-linearity, and re-encrypts under the next node's public key. The re-encryption step can be made non-interactive using proxy re-encryption [59] .

The decryption in the middle of that sentence is the crux of the architecture and was understated in an earlier version of this review. Non-linear activations such as GELU and SwiGLU are not inner products and cannot be evaluated by an inner-product functional key. Applying them requires plaintext, so the pipeline necessarily contains one decryption per node. Describing this architecture as end-to-end encrypted would be false, and the caption of Figure 10 has been corrected accordingly.

What the architecture does provide is compartmentalization: node k sees the activations at layer k and nothing upstream or downstream. Whether that is worth anything depends entirely on how much layer-k activations reveal. Against an adversary who knows the weight matrices, which is the realistic assumption for an open-weight model or a node that has been issued the weights in order to compute, layer-k activations permit approximate inversion back to the input for early layers, and the literature on model inversion attacks [35] applies directly. The compartmentalization is therefore meaningful for deep layers of a proprietary model and weak for early layers of a known one. This is a materially weaker property than the term encrypted inference normally connotes, and system designers should evaluate it as such. An architecture requiring genuine end-to-end encryption of the forward pass must use FHE, at the overhead documented in Section 4.5, and cannot use inner-product FE alone.

6.4.3 *Comparison with alternative approaches*

Table 11 compares FE-based decentralized inference against two alternative privacy-preserving inference paradigms.

Table 11. Comparison of privacy-preserving LLM inference approaches

Approach	Input privacy	Model privacy	Overhead
Hybrid HE/garbled-circuit framework (GAZELLE [49])	Full	Partial	$10^3 - 10^4\times$
Trusted Execution Env. (TEE)	Full	Full	$2 - 10\times$
Split Learning [13], [61]	Partial (leaks intermediate activations to compute nodes)	Partial	$1.1\times$
Lattice FE (this paper)	Partial (per-layer activations)	Partial (per-key)	$10^2 - 10^3\times$

FE occupies an intermediate niche, though the case for it is weaker than an earlier version of this review asserted. It is not true that a compute node cannot reconstruct hidden states without solving LWE: by Proposition 6.2, a node issued a spanning set of functional keys recovers the hidden state by linear algebra. The advantage of FE over split learning is therefore not an absolute confidentiality guarantee but a controllable one. Split learning leaks whatever the cut layer happens to expose, with no mechanism for the model owner to bound it. FE makes the leakage an explicit function of the key-issuance policy, so the system designer can trade node capability against residual uncertainty by choosing the rank allocation. That control is real and useful, and it is a considerably more modest claim than cryptographic hiding. Against full MPC, FE remains cheaper and more interpretable, at a weaker security model. SplitML [13] provides a concrete reference implementation of privacy-preserving federated split learning that combines FHE with differential privacy as a defense-in-depth architecture, demonstrating the viability of the split paradigm for heterogeneous environments while remaining closer to the privacy guarantees of the FE-based approach than classical split learning alone.

6.5 *Engineering deployment considerations*

For systems engineers evaluating lattice-based FE for production inference pipelines, three practical constraints dominate the deployment decision.

6.5.1 *Key management at model scale*

A transformer with $d_{\text{model}} = 4,096$ and $L = 2,048$ context length requires $d_{\text{model}} \times L = 8.4 \times 10^6$ functional keys per attention layer. Each ALS key is a vector of m entries, stored modulo q, and the security proof of requires $m \geq 4n \log_2 q$. At $n = 1,024$ and $q = 2^{32}$ that is $m = 131,072$ entries at 4 bytes each, so one key occupies 512 KiB and the layer requires ≈ 4 TiB. An earlier version of this analysis assumed $m \approx 2n$ and reported 64 GiB; that choice of m is a factor of 64 below what the ALS proof admits, and the corrected figure is the one used throughout this section. The figure should be read as a floor rather than an estimate, for two reasons. The norm bound ALS places on $\boldsymbol{Z}$ implies $\log_2 q \gtrsim 81$ at $n = 1,024$, which is not satisfied by $q = 2^{32}$; solving the parameters self-consistently pushes the layer requirement into the tens of tebibytes. And n and q are modelling choices here, since ALS supplies no concrete instantiation. Structured FE constructions with compact keys are therefore not an optimisation but a precondition for scaling beyond toy models. Because identifying the bottleneck without discussing remedies is of limited use to a systems engineer, Section 6.6.1 treats the available approaches.

6.5.2 *Approaches to the key-size bottleneck*

Four directions address the 4 TiB figure above. None eliminates the problem; each trades a different resource, and the trade is what a designer needs to see.

6.5.3 *Structured key matrices*

The ALS key $\mathrm{sk}_{\boldsymbol{y}} = \boldsymbol{Z}^\top \boldsymbol{y}$ is dense because $\boldsymbol{Z}$ is sampled uniformly. Moving the construction from plain LWE to RLWE replaces that dense matrix with ring elements, so a key becomes a polynomial rather than a full vector over $\mathbb{Z}_q^m$.

This is not hypothetical: Bermudo Mera et al. [62] give the first IPFE scheme whose security rests on RLWE rather than on plain LWE or DDH, the assumptions underlying the earlier inner-product constructions of Abdalla et al. [63] and ALS [25]. It obtains exactly the compactness and performance gains over the LWE construction that the key-size analysis above calls for, and additionally supply compilers that lift a single-input scheme to identity-based and decentralized multi-client variants with ciphertext size linear in the number of clients. For the architecture of Section 6.4 this is the most directly applicable result in the literature, and any serious attempt at the 4 TiB problem should start there rather than from the ALS construction analysed here.

The cost is that structure narrows the security reduction. The resulting assumption is RLWE rather than plain LWE, and although the reduction in that work is carefully constructed, through a multi-hint extended Ring-LWE variant and a ring analogue of the leftover hash lemma, the two assumptions are not equivalent: a break of the ring variant would not require a break of plain LWE.

6.5.4 *Identity-based and attribute-based FE*

Under identity-based FE, a node derives its own key from a public identity string and a single master public parameter, so the authority transmits $O(I)$ material rather than one key per functional. Attribute-based variants generalize this to policies over node attributes, which fits naturally onto the role structure of a deployment where nodes are grouped by trust tier. The cost is that key derivation is now delegated, so revocation becomes the hard problem: withdrawing a node's capability requires either re-keying the master parameter, which invalidates every derived key, or a revocation list that reintroduces per-node state. For deployments with stable node membership this is a favourable trade; for elastic cloud scaling it is not.

6.5.5 *Function-hiding and reusable keys*

If the same projection is applied at every token position, issuing one key per position is wasteful. Reusing a single key across positions reduces the count from $O(d_{\text{model}} \times L)$ to $O(d_{\text{model}})$, removing the context-length factor entirely and cutting the 4 TiB figure by roughly three orders of magnitude at $L = 2,048$. The security consequence must be stated: reuse across positions means a node observing many positions accumulates many evaluations under one key, which accelerates the rank saturation of Proposition 6.2. Key reuse and rank-limited issuance are therefore in direct tension, and the resolution is necessarily a per-deployment calculation rather than a general recommendation.

Table 12 summarises the four directions and the reduction each achieves against the 4 TiB baseline.

Table 12. Approaches to functional key material at transformer scale. Baseline is the 4 TiB per attention layer computed above for $d_{\text{model}} = 4,096$, $L = 2,048$, $n = 1,024$, $q = 2^{32}$ and $m = 4n\log_2 q$. Reduction factors are analytic consequences of the construction, not measured results

Approach	Key material	Factor	Cost incurred
ALS baseline	4 TiB	$1\times$	None; infeasible
Structured $\boldsymbol{Z}$	~ 41 GiB	10^2	Structured-LWE assumption
Identity-based	$O(1)$ master	10^6	Revocation becomes hard
Key reuse across L	2 GiB	2×10^3	Faster rank saturation
Hierarchical	$O(\text{domains})$	varies	Coordinators become trusted

6.5.6 *Hierarchical delegation*

A two-level scheme in which the model owner issues a small number of master keys to trust-domain coordinators, which in turn derive node-specific keys, reduces the authority's transmission to the number of domains rather than the number of nodes. This is the approach that scales best with node count and is developed as a concrete strategy in Section 6.7.

6.5.7 *Network and memory overhead budget*

The re-encryption step between nodes introduces a bandwidth overhead of $2n \log_2 q$ bits per activation vector. For $n = 1,024$, $q = 2^{32}$, and an activation dimension of 4,096, each inter-node transfer is ≈ 32 MB per token, versus ≈ 16 KB for plaintext. This $2,000\times$ bandwidth amplification must be evaluated against the available interconnect before adopting FE-based split inference in a data-center or edge-computing setting.

6.5.8 *TRL and engineering readiness*

The ALS inner-product FE construction [25] is at TRL 3 for LLM workloads: mathematically sound, implemented in research prototypes (e.g., LibSFE), but not validated at transformer scale. The MCFE extension for attention is at TRL 2. The engineering roadmap to TRL 5 requires: compact key generation, GPU-accelerated NTT for FE encryption and decryption, and fused re-encryption kernels for inter-node transfers.

6.5.9 *Collusion resistance: a layered compromise*

Colluding nodes pool their functional keys, so the rank bound of Proposition 6.2 must hold for coalitions rather than individuals: for a tolerated collusion threshold t, every set of t nodes must jointly hold keys of rank strictly less than d. Fully collusion-resistant multi-input FE [64] enforces this cryptographically, and its overhead places it far outside the reach of LLM-scale inference. A practical deployment needs something in between, and the following three-tier strategy is offered as a concrete compromise rather than as a cryptographic solution.

6.5.9.1 *Tier 1: Rank budgeting within trust domains*

Partition the N compute nodes into D trust domains of size N/D, chosen so that nodes within a domain share an operator, jurisdiction, or failure mode and are therefore assumed to collude freely. Allocate functional keys so that the rank available to an entire domain is capped at $r_{dom} < d$. Collusion within a domain is then free but harmless, which is the right assumption, since co-located nodes under one operator should not be treated as independent. This converts an unenforceable assumption about N independent parties into an enforceable one about D administrative boundaries, and D is typically small enough to reason about.

6.5.9.2 *Tier 2: Cross-domain noise flooding*

Cross-domain collusion is the case that cannot be excluded by allocation alone, since the union of two domains may span the space. Here, add calibrated noise to released inner products, with magnitude scaled to the residual uncertainty the allocation is intended to preserve. Unlike the FHE case of Section 4.4, this noise is applied to a released output in the correct space and can be given a differential-privacy interpretation with respect to the input hidden state. The cost is inference accuracy, and the appropriate operating point depends on the task's tolerance, which for classification is typically higher than for generation.

6.5.9.3 *Tier 3: Detection rather than prevention*

Accept that a sufficiently large cross-domain coalition succeeds, and make it detectable and attributable instead. Issue each node keys derived with a distinct pseudorandom tweak, so that a reconstructed hidden state recovered from a coalition carries a traceable signature of which key set produced it, in the manner of traitor tracing. This shifts the defence from cryptographic to contractual and forensic, which is the correct engineering posture when the cryptographic option costs orders of magnitude more than the asset is worth. Table 13 sets out the three tiers and what each costs.

Table 13. Layered collusion strategy. Overhead multipliers are relative to unprotected FE inference and are analytic estimates. Full multi-input FE is included for reference as the cryptographically complete option

Tier	Mechanism	Protects against	Overhead
1	Rank budgeting	Intra-domain collusion, any size	$1\times$
2	Noise flooding	Cross-domain, accuracy cost	$1.1\times$
3	Traced keys	Post hoc attribution only	$1.2\times$
-	Multi-input FE [64]	All coalitions, provably	$10^4\times$

The combined tiers cost roughly $1.3\times$ against the $10^4\times$ of the provable construction, and they buy a materially weaker guarantee. That is the honest characterization: this is a risk-management architecture, not a cryptographic one, and it is appropriate only where the threat model admits administrative and legal controls alongside technical ones.

6.5.10 *Open challenges*

1. Non-linear functions. IPFE/MCFE support linear functions. Softmax and LayerNorm require either a trusted computation step, a polynomial approximation, or a switch to FHE for those layers, breaking the clean FE model.

2. Key size and management. The number of functional keys scales as $O(d_{\text{model}} \times L)$ for attention, which is enormous for large models ($d_{\text{model}} = 4,096$, $L = 4,096$). Structured (hierarchical or compact) FE constructions are needed.

3. Re-encryption overhead. Encrypting intermediate activations before forwarding to the next node adds latency proportional to the number of nodes and the activation dimension.

4. Collusion resistance. If multiple nodes collude, they may combine their functional keys to reconstruct the full input. Collusion-resistant multi-input FE [64] is known but not yet efficient enough for production use.

5. Decentralized key issuance. The current model assumes a trusted model owner. Decentralizing key issuance via threshold FE removes this single point of trust but increases protocol complexity.

7. Lattice methods among competing privacy technologies

A survey confined to lattice-based methods risks implying that they are the natural choice for privacy-preserving AI. For most workloads deployed today they are not. An engineer with a confidentiality requirement and a budget chooses among at least six families, and lattice-based encryption is usually the most expensive of them. This section states what the alternatives provide, and where the lattice approach is and is not the right answer.

7.1 *The alternatives*

7.1.1 *Trusted execution environments*

A TEE such as Intel Trust Domain Extensions (TDX), AAdvanced Micro Devices (AMD) Secure Encrypted Virtualization–Secure Nested Paging (SEV-SNP), or NVIDIA confidential computing on Hopper and later GPUs executes computation in hardware-isolated memory that the host operating system and hypervisor cannot read. The overhead is far smaller than the other entries in this list and is worth stating precisely, because it is the number against which every cryptographic alternative must justify itself. Zhu et al. [65] benchmark TEE mode on NVIDIA Hopper GPUs across a range of models and token lengths and report average throughput overhead below 9%, with a worst observed case near 9% and with larger models and longer sequences approaching zero overhead. Latency is affected more than throughput: time to first token degrades by up to roughly 26% in their measurements. These figures are drawn from a preprint that has not completed peer review, and should be read as preliminary. The residual cost is attributable almost entirely to encrypted CPU-GPU transfers over PCIe rather than to computation inside the GPU, which is why it amortizes away as sequence length grows: the transfer is fixed while the compute is not. A cryptographic scheme competing with this must close a gap of three to four orders of magnitude. The security model is the decisive difference: trust is placed in a hardware

vendor and an attestation chain rather than in a mathematical assumption. Misono et al. [66] provide the systematic evaluation of SEV-SNP and TDX that this comparison requires, covering both performance across the memory, storage, and attestation paths and a trusted-computing-base and Common Vulnerabilities and Exposures (CVE) analysis. The record of side-channel attacks against enclave technologies is substantial and continuing, and a compromise is a total loss of confidentiality rather than a graceful degradation. TEEs also do not protect against a malicious vendor or a compelled disclosure at the hardware root.

7.1.2 *Secure multi-party computation*

MPC distributes computation across parties so that no proper subset learns the inputs, with security from secret sharing and oblivious transfer rather than from a hardness assumption on a single ciphertext. Transformer-specific protocols have advanced quickly: Iron [43] supplies protocols for matrix multiplication together with softmax, GELU, and LayerNorm, reporting three to fourteen times less communication than prior art, and PUMA [44] evaluates LLaMA-7B in roughly five minutes per generated token, the first demonstration at that parameter scale under MPC. That figure is the most useful single data point in this section: it is slow enough to rule out interactive use and fast enough that batch processing of sensitive documents is conceivable, which is a materially different position from FHE-based training.

Overhead for neural network inference is typically 10^2 to $10^4\times$, comparable to FHE, but the cost profile differs in a way that matters for deployment: MPC is communication-bound rather than compute-bound. Round complexity scales with circuit depth, so latency is dominated by network round trips. MPC is well suited to a low-latency datacentre interconnect and poorly suited to wide-area or edge deployment, which is close to the opposite of the FHE profile.

7.1.3 *Federated learning*

FL keeps raw data on client devices and shares only model updates. Overhead relative to centralized training is small, often under 1.5 ×, and it is widely deployed. It provides no cryptographic guarantee on its own: gradient inversion attacks reconstruct training examples from shared updates, particularly for small batches. Liu et al. [67] provide a systematic analysis of reconstruction attacks, formulating reconstruction as an inverse problem and deriving theoretical bounds on reconstruction error, which is a firmer basis for comparing defences than the anecdotal attack demonstrations that preceded it. Notably for the present survey, they find gradient pruning the most effective defence among those evaluated. FL is best understood as a data-governance architecture that reduces the blast radius of a breach, and it requires composition with secure aggregation or differential privacy to make a defensible privacy claim.

7.1.4 *Split learning*

SL partitions the network between client and server at a cut layer, with the client computing the early layers and sending activations onward. Overhead is near unity. Erdoğan et al. [68] show that an honest-but-curious server knowing only the client architecture can invert the shared activations to recover input samples and additionally steal a functionally equivalent client model, without the client detecting it. That result is the reason this review treats split learning as a governance measure rather than a confidentiality mechanism, and the same reasoning applies to the functional-encryption pipeline of Section 6.4, which exposes activations at every node boundary. The privacy depends entirely on how much the cut-layer activations reveal, and for early cut layers with known weights this is a great deal. It shares the structural weakness identified for FE-based inference in Section 6.4, and for the same reason.

7.1.5 *Differential privacy*

DP is not an alternative to the others but an orthogonal axis, and the confusion of these two things is the error corrected in Section 4.4. DP bounds what the released output reveals about any individual record. It composes with every technology in this list and substitutes for none of them. Its cost is model accuracy rather than compute, which makes it uniquely attractive when a utility loss is tolerable and uniquely useless when it is not.

7.1.6 *Confidential computing as an assembly*

What is marketed as confidential computing is generally a TEE combined with attestation, encrypted memory, and encrypted transport. It is an integration pattern rather than a distinct primitive, and its properties are those of the underlying TEE.

7.1.7 *Comparative assessment*

Table 14 sets the six families against the criteria a systems engineer actually applies. The overhead figures are order-of-magnitude indicators drawn from published benchmarks (evidence class E2) and vary by more than a factor of ten with model architecture, parameter selection, and hardware, so they should be used to rank options rather than to size a deployment.

Table 14. Privacy-enhancing technologies for LLM workloads. Overheads are relative to unprotected plaintext computation and are order-of-magnitude indicators only (evidence class E2). Maturity is expressed as Technology Readiness Level under the scale defined in Table 17

Technology	Trust rests on	Infer.	Train	Principal limitation	TRL
TEE/confidential computing	Hardware vendor, attestation	$1.0 - 1.09\times$ (throughput; up to $1.26\times$ latency) [65]	$1.2 - 3\times$	Side-channel exposure; vendor is in the trust base	9
MPC	Non-collusion of parties	$10^2 - 10^4\times$	$10^3 - 10^5\times$	Communication-bound; round complexity scales with depth	5
FHE	LWE hardness	$10^3 - 10^4\times$	$10^4 - 10^6\times$	Non-polynomial circuits; bootstrapping cost	4
Functional encryption	LWE hardness	$10^2 - 10^3\times$	Not applicable	Linear functionality only; key size; span attacks	3
Federated learning	Client honesty, aggregator	$1\times$	$1.1 - 1.5\times$	Gradient inversion; no guarantee unaided	9
Split learning	Cut-layer opacity	$1.1\times$	$1.2\times$	Activation leakage; inversion for early cuts	7
Differential privacy	Nothing: information-theoretic	$1\times$	$1 - 2\times$	Accuracy cost; orthogonal, not substitutable	9

7.2 *Normalized resource comparison at 128-bit security*

Reviewers of an earlier version observed, correctly, that latency, memory, and bandwidth figures were scattered across sections at incommensurable parameter settings. Table 15 fixes a single workload and reports all three axes against it. The workload is one forward pass of a single transformer block at $d_{\text{model}} = 768$, sequence length 128, batch size 1, with all cryptographic parameters calibrated to 128-bit classical security.

Three observations follow from the table, and they are the practical takeaways of this section.

First, the memory column is more binding than the latency column, and it is the one usually omitted from comparisons. An FHE working set of tens of gigabytes for a single block of a small model exceeds the memory of most inference accelerators before any consideration of throughput. Latency can be addressed with more hardware; a working set that does not fit cannot.

Second, MPC and FHE fail in different environments. MPC's dependence on round trips makes it viable inside a datacentre and unusable across a wide-area link, while FHE's compute-bound profile is indifferent to network topology but demands hardware that does not yet exist at the required price. The choice between them is therefore largely determined by deployment topology rather than by security preference.

Third, the honest ranking for a practitioner today places TEEs first for production LLM inference, with MPC viable for latency-tolerant datacentre workloads, and FHE and FE confined to research and to narrow high-value cases where

the trust model genuinely excludes hardware vendors. The argument for lattice methods is not that they are currently competitive. It is that they are the only options in this list whose security does not degrade when the adversary gains physical access, and the only ones that remain sound if a hardware root of trust is compromised. For data with a decades-long confidentiality lifetime, that distinction may eventually justify the cost. It does not justify it today for most workloads, and this review does not claim otherwise.

Table 15. Resource cost of one transformer block forward pass ($d_{\text{model}} = 768$, $L = 128$, batch 1) at 128-bit security. Figures are derived from published benchmarks and from the parameter arithmetic in Sections 4.5 and 6.6; they are engineering estimates for comparison, not measurements of a single system. Bandwidth is per block per token where applicable

Technology	Latency	Memory	Bandwidth	Bound by
Plaintext (baseline)	$\sim$ 1 ms	50 MB	3 KB	Compute
TEE	1-2 ms	55 MB	3 KB	Compute
Split learning	1-2 ms	50 MB	200 KB	Network
MPC (2-party)	0.5-5 s	2-8 GB	50-500 MB	Network rounds
FHE (CKKS)	2-20 s	8-40 GB	30-100 MB	Compute, bootstrapping
Functional encr.	0.1-1 s	4-70 GB	30 MB	Key material

8. Comparative taxonomy

Synthesizing three research frontiers into a coherent framework requires identifying both what they share and where they diverge. The NIST AI Risk Management Framework [30] organizes trustworthiness requirements for AI systems into dimensions including confidentiality, integrity, explainability, and resilience. Each of the three frontiers in this survey addresses a different subset of these dimensions: FHE-protected training targets confidentiality and integrity of training data, LLM-guided cryptanalysis stress-tests the resilience of the underlying hard problems, and FE-based inference provides fine-grained, provably bounded information disclosure. This section maps those concerns to the algebraic structure of lattice-based cryptography to show why LBC is uniquely suited to addressing all three simultaneously.

8.1 *Unified framework*

Table 16. Comprehensive taxonomy of LBC-LLM research frontiers

Dimension	F1: FHE Training	F2: LLM Cryptanalysis	F3: FE Inference
Role of LBC	Privacy provider	Attack target	Access controller
Role of LLM	Learner on encrypted data	Heuristic solver	Computation target
Hard problem	RLWE (CKKS/BGV)	LWE/SVP (to break)	LWE (IPFE/MCFE)
Noise role	Obstacle (limit depth)	Target (remove to win)	Shield (hide input)
Threat model	Malicious compute server	Adversary with oracle access	Curious compute node
Security goal	Data confidentiality	Security analysis	Input/weight privacy
LLM interaction	Passive (encrypted training)	Active (guides solver)	Passive (encrypted inference)
Overhead	$10^3 - 10^4\times$	Adds LLM query cost	$10^2 - 10^3\times$
Maturity	Prototype (small LLMs)	Conceptual/preliminary	Theoretical/preliminary
Nearest prior work	CrypTen, Delphi	FunSearch, NeuroSAT	GAZELLE, split-learning

The three frontiers share a common structural tension: they all exploit the noise structure inherent in lattice cryptography, but for diametrically opposite purposes. In FHE-based training, noise is an obstacle to computational precision that must be managed. In FE-based inference, noise provides the ciphertext indistinguishability that enables fine-

grained access control. In LLM-guided cryptanalysis, noise is the target: reducing the noise budget to zero is equivalent to recovering the secret.

Table 16 provides a comprehensive taxonomy across the three frontiers along six dimensions.

8.2 *Cross-cutting themes*

Four structural themes cut across all three frontiers regardless of the specific cryptographic scheme or LLM architecture involved. Understanding these themes is more useful than studying each frontier in isolation, because progress on any one of them simultaneously unlocks advances across all three research directions.

8.2.1 *Noise budget management*

Noise is the unifying variable across all three frontiers, playing a different role in each but remaining central to every security argument. All three frontiers must reason carefully about noise. In FHE training, the noise budget determines how many gradient steps can be performed before bootstrapping is required. In FE inference, the error term must remain small enough that Dec succeeds on every layer output. In cryptanalysis, estimating the noise budget is precisely the LWE distinguishing problem that BKZ attacks. A unified framework for noise budget analysis across all three settings would be a significant theoretical contribution.

8.2.2 *The security-utility tradeoff*

Each frontier exhibits a characteristic security-utility tradeoff:

• **F1 (FHE Training):** Larger noise parameter σ improves ciphertext security and membership inference resistance but reduces gradient SNR and slows convergence.

• **F2 (Cryptanalysis):** A more capable LLM heuristic improves attack efficiency but also enables practitioners to set more conservative security parameters.

• **F3 (FE Inference):** Issuing fewer functional keys reduces information leakage but limits the expressiveness of the function class a node can evaluate.

8.2.3 *Polynomial approximation as a recurring challenge*

Both F1 and F3 require approximating non-polynomial functions (softmax, GELU, LayerNorm) within lattice-based schemes. The degree and precision of these approximations determine the compute-security tradeoff. This challenge is not unique to LLMs: it appears in any FHE or FE application to neural networks [42], [49] . However, the scale of modern LLMs ($d_{\text{model}} = 4,096 - 16,384, L = 4,096 - 128,000$) makes the problem qualitatively harder than in prior work on smaller CNN inference.

8.2.4 *Decentralization and trust assumptions*

All three frontiers have variants that can be decentralized, removing the need for a single trusted party:

• F1: Multi-key FHE enables multiple data owners to contribute to training without a shared secret key.

• F2: The LLM policy can be distributed across multiple agents to prevent a single attacker from controlling the full reduction strategy.

• F3: Threshold FE distributes key generation so no single model owner holds the master secret key.

8.3 *Research interaction graph*

The three frontiers are not independent: advances in one directly affect the others. Figure 11 illustrates the key dependencies.

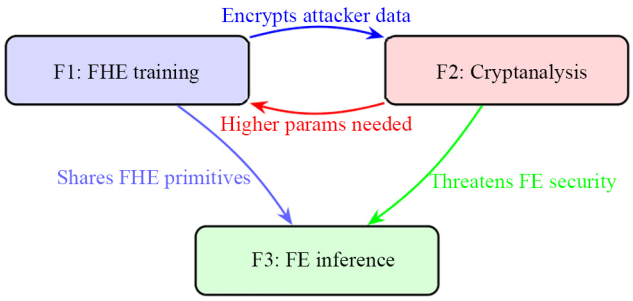

Figure 11. Research interaction graph among the three frontiers. Directed edges indicate that progress in the source frontier enables or threatens progress in the target frontier

A directed graph shows which frontiers touch which, but not whether the coupling helps or harms, and a reader cannot act on an arrow alone. Each edge is therefore given its sign and its mechanism below. Signs are stated from the perspective of a deployer of privacy-preserving AI: positive means progress in the source frontier makes that deployer's position better.

8.3.1 *F2 → F1, F3: negative, and weaker than it appears*

A cryptanalytic advance that lowered the effective security of lattice schemes would force parameter increases across every FHE and FE deployment, raising overhead superlinearly, since larger n increases both ciphertext size and per-operation cost. This is the coupling that motivates treating cryptanalysis in a survey otherwise concerned with construction. Its practical weight is however limited by Proposition 5.1: adaptive scheduling cannot lower core-SVP cost, so this edge represents a risk channel that the specific technique of Frontier 2 does not currently activate.

8.3.2 *F1 → F2: positive for capability, negative for security, and symmetric in an uncomfortable way*

Cheaper homomorphic computation makes it more practical to train models on sensitive data, including models trained to attack lattices. The same efficiency gain that protects a hospital's records also lowers the cost of building the adversary. This symmetry is intrinsic and not resolvable by technical means.

8.3.3 *F1 ↔ F3: strongly positive, shared substrate*

Both depend on the number-theoretic transform and on polynomial approximation of non-linearities. An NTT accelerator or a better minimax approximation for GELU benefits both at once, which is why the hardware argument of Section 9 is stronger than it would be for either frontier alone. This is the most reliable edge in the graph and the one with the clearest investment case.

8.3.4 *F3 → F1: positive, bounded*

Functional encryption can discharge the linear portions of a pipeline cheaply, leaving FHE for the non-linear remainder. The bound on this edge is Section 6.3.1: since attention is bilinear and outside the IPFE class, the division of labour cannot extend to the component that dominates transformer cost.

8.3.5 *Common-mode risk: all three, negative*

The edge most often omitted is the one from the shared assumption to all three frontiers simultaneously. Because F1, F2, and F3 all rest on the hardness of structured lattice problems, a break does not degrade one frontier; it removes the foundation of the entire research programme described in this review. Diversification, if it is wanted, must come from outside this survey's scope, which is the reasoning behind the NIST decision to standardize a code-based mechanism alongside ML-KEM.

8.4 *Maturity assessment*

Assessing research maturity requires separating theoretical soundness from engineering readiness. The NIST AI RMF Govern function [30] calls for organizations to understand the risk posture of AI technologies before adoption, which for lattice-protected LLMs means distinguishing between cryptographic guarantees that are provable today and system-level properties that remain aspirational. The Technology Readiness Level (TRL) scale captures this distinction, but only if the scale is defined. Reviewers of an earlier version noted, correctly, that TRL values were assigned without stated criteria and were consequently not falsifiable. Table 17 therefore adapts the generic nine-level scale to cryptographic systems, giving each level an observable admission test. A level is claimed only when the test in its row has demonstrably been met by published work.

Table 17. TRL scale as specialized to cryptographic AI systems in this review. Each level carries an observable admission test; a claim at level k asserts that the tests for all levels up to k have been met by published work. This is the scale used in Tables 14 and 18

TRL	Stage	Admission test
1	Principle observed	A security or functionality goal is stated in formal terms
2	Concept formulated	A construction is proposed, with a security definition it is intended to meet
3	Proof of concept	Security proof under a stated assumption, plus a reference implementation at toy parameters
4	Validated in lab	Implementation at cryptographically meaningful parameters, correctness verified, performance measured
5	Validated in relevant environment	Runs on a realistic workload, not a microbenchmark; independent reimplementation exists
6	Demonstrated in relevant environment	End-to-end system on representative data with measured accuracy and latency
7	Prototype in operational environment	Deployed on real traffic with real users, non-production
8	Qualified	Production deployment; security audit or standardization review completed
9	Proven in operations	Sustained production use at scale across multiple independent operators

Applying this scale yields the assignments in Table 18. Two of the three frontiers move down relative to the earlier version of this review, which is the expected consequence of requiring evidence for each level rather than assigning by impression.

Table 18. TRL assignments with justification. The blocking test is the lowest unmet admission test from Table 17, which is the quantity a research programme should target

Item	TRL	Highest test met	Blocking test
Encrypted inference, small models	4	Measured at real CKKS parameters on classification workloads	Level 5: no independent reimplementation on a representative LLM workload
Encrypted LoRA fine-tuning (F1)	2	Protocol proposed with security intent	Level 3: no reference implementation; circuits in Section 4.3.2 unresolved
LLM-guided lattice reduction (F2)	1	Goal stated formally	Level 2: no construction with a stated success criterion; and Proposition 5.1 bounds the achievable outcome
ALS inner-product FE (F3)	3	Security proof plus toy implementation (Figure 12)	Level 4: key material of 4 TiB per layer precludes cryptographic-scale implementation
FE transformer inference (F3)	1	Goal stated	Level 2: attention is outside the IPFE functionality class (Section 6.3.1)
NTT hardware acceleration	6	FPGA and ASIC designs demonstrated on representative kernels	Level 7: not deployed in operational AI inference paths

The downgrade of F2 from the previously claimed TRL 1-2 to TRL 1 is the most consequential change and follows directly from the admission test for level 2: a concept is formulated only when accompanied by a criterion for success. Section 5.6 now supplies such a criterion in the form of a benchmark protocol, so an experiment executing that protocol, whatever its outcome, would establish level 2.

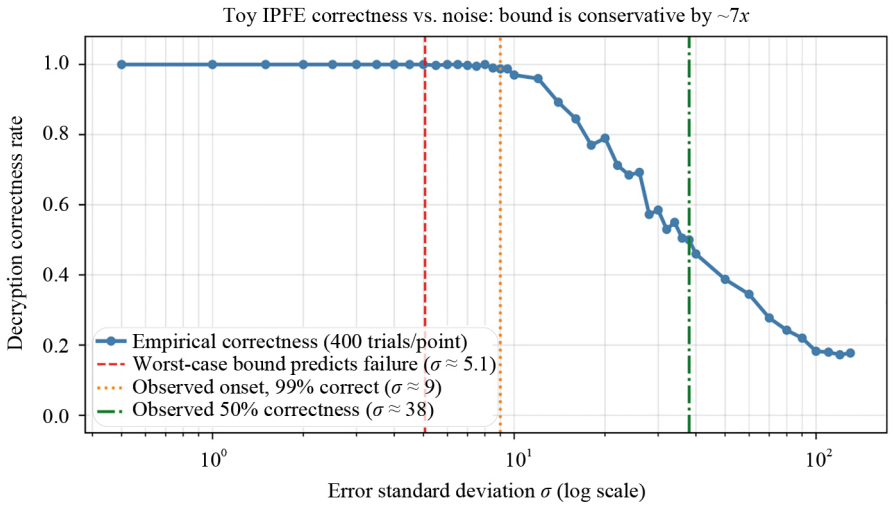

Figure 12. Empirical correctness of the toy LWE-IPFE construction against error standard deviation σ, on a logarithmic σ axis ($n = 10$, $m = 40$, $\ell = 8$, $q = 131,072$, $B = 256$, entries of $\boldsymbol{x}$ and $\boldsymbol{y}$ drawn uniformly from $\{-2, \ldots, 2\}$, 400 trials per point). The worst-case bound $\sigma \approx \lfloor q/(2B) \rfloor / (\sqrt{n\ell}\,\|\boldsymbol{y}\|_\infty) \approx 5.1$ predicts failure roughly seven times earlier than observed: correctness remains at unity until $\sigma \approx 9$ and reaches one half only near $\sigma \approx 38$. The discrepancy is the point of the figure. It arises because the bound compounds worst-case magnitudes across independent terms, whereas the realised inner-product noise concentrates far below that envelope. An earlier version of this review described a sharp degradation at $\sigma \approx 5.1$; no such cliff exists in the data, and engineers calibrating noise for a deployed IPFE instance should measure rather than rely on the analytic bound, which is conservative by close to an order of magnitude. Evidence class E3: toy parameters, illustrative of qualitative behaviour only

9. Deployment realities

The preceding sections analyse constructions. This section addresses what happens when one is put into a production environment, which is where most of the surveyed techniques currently fail for reasons that have nothing to do with cryptography. The figures below are engineering estimates derived from the parameter arithmetic of earlier sections and from published benchmarks; they carry evidence class E2 and are intended to support order-of-magnitude decisions.

9.1 *Memory is the binding constraint*

Discussion of FHE cost conventionally centres on latency, which is the wrong emphasis. Latency can be bought down with parallelism; a working set that exceeds accelerator memory cannot be.

Consider CKKS at $N = 2^{16}$ with a 1,200-bit modulus chain, which is representative of parameters supporting the multiplicative depth that a transformer block requires. A single ciphertext occupies roughly $2 \times N \times \log_2 q / 8 \approx 20$ MB. One transformer block at $d_{\text{model}} = 768$ with a sequence length of 128 requires on the order of several hundred such ciphertexts live simultaneously once activations, weights, and intermediate products are counted, before the evaluation keys. Rotation keys alone, which are needed for the rotate-and-sum matrix products, run to several gigabytes at these parameters and must be resident.

The resulting working set of tens of gigabytes per block sits at or above the capacity of current inference accelerators, which is why encrypted transformer inference demonstrations are confined to small models and short sequences. The practical implication is that memory capacity, not arithmetic throughput, sets the model size ceiling, and an accelerator design that optimizes NTT throughput without addressing capacity and bandwidth will not move that ceiling.

9.2 *Communication and topology*

FHE and MPC fail in opposite environments, and this determines deployment topology more than any security consideration.

MPC's cost is dominated by round trips proportional to circuit depth. On a datacentre interconnect with sub-millisecond round-trip time this is tolerable; across a wide-area link at 50 ms it is not, since a transformer block's depth multiplied by that latency exceeds any interactive budget. MPC is therefore effectively a single-datacentre or single-availability-zone technology.

FHE requires no interaction during evaluation, so it is indifferent to network latency and suits asynchronous, wide-area, and edge topologies. Its bandwidth cost is nonetheless substantial: the ciphertext expansion factor of 10^3 to 10^4 means that transmitting an encrypted prompt and receiving an encrypted response moves tens of megabytes where plaintext would move kilobytes. For a mobile client this is prohibitive on a metered connection and marginal even on broadband.

Functional encryption inherits the worst of both in the decentralized architecture of Section 6.4, since it requires both large key material distributed in advance and per-hop re-encryption traffic between nodes.

9.3 *Energy*

Energy consumption is rarely reported in the FHE literature and is a material consideration for any operator at scale. A first-order estimate follows from the overhead multiplier: if encrypted inference costs 10^3 to 10^4 times the arithmetic of plaintext inference, and if the workload remains compute-bound, energy per query scales similarly. At a plaintext inference cost on the order of a joule per query for a small model, encrypted inference lands in the kilojoule range, which is a difference between a negligible and a non-negligible line item at high query volume.

This estimate is optimistic in one respect and pessimistic in another. It is optimistic because data movement, not arithmetic, dominates energy in modern accelerators, and FHE's poor arithmetic intensity relative to its memory footprint makes it worse than the multiplier suggests. It is pessimistic because dedicated hardware improves energy efficiency by more than it improves latency, since much of the general-purpose overhead is eliminated. The net effect has not been measured, and a careful energy characterization of homomorphic inference would be a useful contribution to this literature.

9.4 *Cloud deployment and operational constraints*

Several operational realities constrain adoption independently of performance.

9.4.1 *Key management is the integration burden*

FHE and FE both require key material that dwarfs anything conventional key management systems are designed for. Evaluation keys of several gigabytes cannot be stored in a standard secrets manager, cannot be held in a hardware security module of typical capacity, and must be distributed to every compute node. Rotating them is an operation measured in hours. This is a mundane obstacle and, in the authors' assessment, a more immediate barrier to adoption than the compute overhead, because it has no hardware solution on any roadmap.

9.4.2 *Debugging is materially harder*

An encrypted pipeline cannot be inspected. When output is wrong, the engineer cannot distinguish a model problem from a noise-budget exhaustion from an approximation error without decrypting, which the threat model may forbid in production. Approximate schemes such as CKKS compound this, since small errors are expected and only their magnitude distinguishes correct operation from failure. Operational tooling for encrypted workloads is essentially absent.

9.4.3 *Regulatory recognition lags the technology*

Compliance regimes generally recognize encryption at rest and in transit. Computation on encrypted data does not map cleanly onto existing categories, so an organization deploying FHE may find it has incurred substantial cost without

a corresponding reduction in compliance obligation. This is a policy gap rather than a technical one, and it materially affects the business case.

9.5 *Hardware co-design*

The case for dedicated hardware is stronger than for most accelerator proposals, because the target is narrow and shared. The number-theoretic transform dominates cost in FHE, in FE, and in post-quantum key exchange alike, so a single accelerator design amortizes across all three, as noted in Section 3.3.

Published FPGA and ASIC designs demonstrate one to two orders of magnitude improvement on NTT kernels, which is real but insufficient on its own: closing a 10^3 to 10^4 gap requires algorithmic improvement of comparable magnitude alongside the hardware. The dedicated-silicon literature is now substantial enough to be surveyed in its own right; Zhang et al. [17] systematize fourteen accelerator designs and provide testbed comparisons across the open-source subset. Among these, CraterLake [69] is one of the most relevant accelerator designs for the workloads considered here, supporting FHE programs with unbounded multiplicative depth through fully packed bootstrapping, and reporting gains of several thousand times over CPU on deep programs including neural networks. The qualification that matters for this review is that such results are reported for accelerators that do not exist as purchasable hardware, so they establish what is architecturally attainable rather than what a systems engineer can specify today.

GPU acceleration is the nearer-term lever and has moved quickly. OpenFHE [70] is the reference CPU implementation against which most work is measured, and FIDESlib [71] reports no less than 70× speedup over the AVX-optimized OpenFHE bootstrapping path while remaining interoperable with OpenFHE on the client side. An improvement of this size on the operation that dominates deep circuits is the most consequential single development for the practicality of the constructions surveyed here, and it is worth being precise about what it does and does not achieve: it reduces a $10^4\times$ gap to a few hundred times, which is the difference between infeasible and merely expensive for inference, and leaves training out of reach. The result is reported in a preprint that has not completed peer review and should be treated as preliminary pending independent replication. The realistic path to viable encrypted inference is multiplicative, combining hardware acceleration, better polynomial approximations reducing required depth, and parameter selection tuned per workload rather than conservatively. No single factor closes the gap.

The most significant unaddressed requirement is memory bandwidth rather than arithmetic throughput. An NTT accelerator attached to insufficient high bandwidth memory will be starved, and the working-set analysis above suggests this is the regime current designs operate in.

9.6 *Industrial adoption status*

Honest characterization of the current state: encrypted machine learning is deployed in production in narrow settings involving small models, low query volumes, and high-value data, principally in regulated finance and healthcare analytics. Encrypted inference for large language models is not in production anywhere known to the authors. Encrypted training of large models has not been demonstrated at any scale beyond research prototypes.

Where confidential AI is deployed at scale today, it uses trusted execution environments, for the reasons set out in Section 7. An engineer with a present-day requirement should evaluate that path first. The lattice-based techniques surveyed here are, at the time of writing, a research programme with a credible long-term rationale rather than an available alternative, and this review would be misleading if it left any other impression.

10. Engineering roadmap and open problems

The three frontiers surveyed in this paper share several structural bottlenecks: the computational gap between FHE overhead and practical training throughput, the absence of empirical evidence for LLM-guided cryptanalytic improvement, and the lack of end-to-end demonstrations of lattice-FE inference at transformer scale.

An unordered list of open problems is of limited use, since it gives no basis for allocating effort. Table 19 therefore ranks the milestones by three factors a research programme actually weighs: technical feasibility given current tools, expected impact if achieved, and the TRL transition the milestone would effect under the scale of Table 17. Each milestone is stated so that its completion is checkable, which is the property the earlier version of this section lacked. Figure 13 shows the dependency structure among them.

The horizons below expand on these milestones. Each is anchored to the TRL scale of Table 17.

Table 19. Prioritized research milestones. Feasibility and impact are on a three-point scale (H, M, L) assigned by the authors; the criteria are stated in the text and the assignments are contestable, which is the intent. Priority is the ordering the authors would follow, and milestone M1 is ranked first because every other item on the list is downstream of it

ID	Horizon	Milestone (completion criterion)	Feas.	Imp.	TRL
M1	1-2 yr	Encrypted LoRA fine-tuning of a 125M-parameter model to within 2% of plaintext accuracy on a standard benchmark, with published code	M	H	$2 \rightarrow 4$
M2	1-2 yr	Polynomial approximations for GELU, softmax, and LayerNorm with certified error bounds valid over ranges observed in real transformer activations	H	H	$4 \rightarrow 5$
M3	1-2 yr	Execution of the benchmark protocol of Section 5.6, publishing the result whether positive or negative	H	M	$1 \rightarrow 2$
M4	2-3 yr	GPU NTT kernel sustaining CKKS bootstrapping at < 100 ms for $N = 2^{16}$, released as open source	M	H	$6 \rightarrow 7$
M5	2-3 yr	Compact-key IPFE reducing per-layer key material below 1 GB at $d_{\text{model}} = 4,096$ with a stated security reduction	M	M	$3 \rightarrow 4$
M6	2-4 yr	Lattice-based decentralized MCFE, replacing the pairing-based construction of [60] with a post-quantum one	L	M	$1 \rightarrow 3$
M7	3-5 yr	Quadratic FE efficient enough for one attention head, addressing Section 6.3.1	L	H	$1 \rightarrow 3$
M8	3-5 yr	Encrypted fine-tuning at 1B parameters within a $10^3\times$ overhead budget on commodity accelerators	L	H	$2 \rightarrow 5$
M9	5+ yr	Standardized parameter sets and security levels for FHE, analogous to the FIPS 203 parameter sets	M	H	$4 \rightarrow 8$

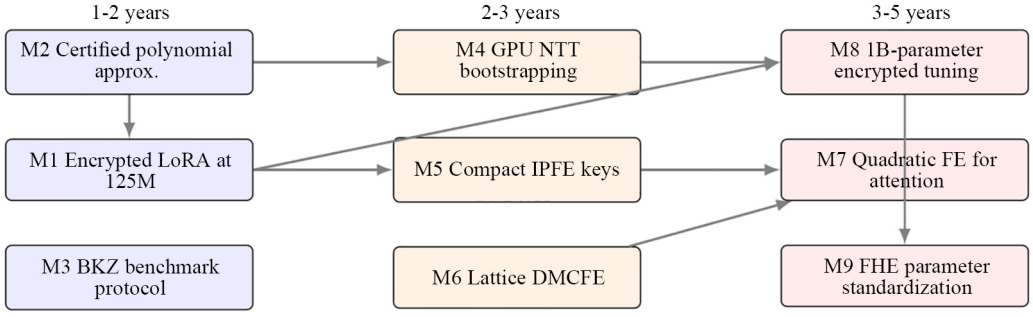

Figure 13. Dependency structure of the roadmap milestones from Table 19. Arrows denote prerequisite relations. M2, certified polynomial approximation with error bounds valid over observed activation ranges, is the root of most of the graph: it gates encrypted fine-tuning, and its absence is the reason the circuits enumerated in Section 4.3.2 remain open. M3 is deliberately disconnected, since the cryptanalytic question is independent of the construction programme and can be settled in parallel by a different group

10.1 *Short-term research priorities (1-3 years)*

Short-term priorities are those where the research challenge is well-defined and a credible technical path to solution exists, but engineering gaps prevent deployment at production database and LLM scales. A useful lens for understanding urgency is the expected-loss model from quantitative risk analysis: a security control is financially justified when its annual cost is less than the expected-loss reduction it provides. At current FHE overhead levels ($10^3 - 10^4\times$ slowdown), the engineering cost of an FHE-protected LLM training run exceeds the expected-loss reduction for most organizations, making overhead reduction the single most impactful line of research. Similarly, the absence of a benchmark suite for LLM-guided lattice reduction makes it impossible to calibrate whether the risk to current NIST standards is negligible or material. And the key management burden of full lattice-FE inference is currently prohibitive at production dimension. Each of the four priorities below directly addresses one of these three blocking constraints.

10.1.1 *Efficient bootstrapping for LLM workloads*

CKKS bootstrapping currently costs ~ 2 seconds per ciphertext on server hardware. A 12-layer transformer requires at least one bootstrapping step per residual block, placing the end-to-end forward pass time at ~ 24 seconds even without considering the attention mechanism. Hardware accelerators specifically designed for the NTT (Number Theoretic Transform) operations that dominate FHE arithmetic are a prerequisite for practical adoption. ASIC designs for FHE are an active research area [48] , [69] , and GPU-accelerated frameworks are emerging.

10.1.2 *Differentially private FHE*

A rigorous composition theorem for FHE noise and differential privacy would clarify exactly what post-decryption guarantees FHE-trained LLMs provide. The key challenge is that FHE security is computational while DP is information-theoretic; a meaningful composition requires either relaxing DP to its computational variant [72], which is defined precisely to quantify over polynomial-time adversaries and is therefore the correct target for a composition with an encryption scheme, or treating the FHE noise as an additive Gaussian mechanism with equivalent sensitivity, which Section 4.4 shows is not available.

10.1.3 *Benchmark suite for LLM-Guided lattice reduction*

The absence of a standardized benchmark makes it impossible to compare LLM-guided BKZ strategies against each other or against non-ML baselines. A benchmark suite should include:

1. Lattice instances at the dimensions specified in Section 5.6, $n \in \{60,\ 80,\ 100,\ 120,\ 140\}$, which span the range where exhaustive comparison against optimal static schedules is still feasible. Dimensions at cryptographic scale cannot serve as a benchmark here, because the optimal static baseline against which a learned schedule must be judged is not computable there.
2. A fixed budget (e.g., total SVP oracle calls) to measure reduction quality per unit cost.
3. Baselines: standard BKZ, progressive BKZ, and random block-size sampling.

10.1.4 *Compact FE for transformer attention*

Current IPFE schemes require one functional key per output dimension. For attention heads with $d_k = 128$ and $H = 32$ heads, this is $32 \times 128 = 4,096$ keys per layer per node. Compact or aggregated FE constructions that bundle multiple inner products into a single key would dramatically reduce key management overhead.

10.2 *Medium-term research priorities (3-5 years)*

The medium-term agenda shifts from fixing individual bottlenecks to co-designing LBC and LLM systems end to end. The four priorities below each require results from both the cryptography and machine learning communities and are unlikely to be resolved by either field in isolation.

10.2.1 *Post-quantum secure split-training protocols*

Current split-learning frameworks [13], [61], use classical public-key encryption for inter-node communication. A post-quantum variant that uses Module-LWE key encapsulation (ML-KEM) for communication and RLWE-based CKKS for computation would provide end-to-end post-quantum security across the entire training pipeline.

10.2.2 *LLM-designed LBC parameters*

Rather than using an LLM to attack existing parameters, one could use an LLM to search for better parameter sets that achieve the desired security level with lower overhead. Concretely, an LLM trained on the LWE estimator codebase could propose novel modulus hierarchies (for RNS-CKKS) or error-correction codes that reduce the noise budget requirement for a given security level.

10.2.3 *FE-Native transformer architectures*

Most LLM architectures were designed for plaintext computation. A new architecture co-designed for FE would likely look different: linear-only attention (e.g., replacing softmax with a linear kernel), removal of LayerNorm in favor of prenormalization with bounded activations, and quantized embedding lookup via BGV/BFV. Such an architecture could achieve end-to-end lattice-secure inference with significantly lower overhead than retrofitting FHE/FE onto standard transformers.

10.2.4 *Security proofs for LLM-guided attacks*

The LLM cryptanalysis frontier currently lacks theoretical grounding. A proof that an LLM-guided strategy cannot improve over optimal non-adaptive BKZ by more than a polynomial factor would significantly strengthen security claims for LWE-based schemes. Conversely, a construction showing super-polynomial improvement would be a landmark result with implications for all post-quantum cryptography.

10.3 *Long-term research vision (5+ years)*

The long-term vision shared by all three frontiers is a computing infrastructure in which the boundaries between plaintext and ciphertext become operationally invisible across the full AI lifecycle. The NIST AI Risk Management Framework [30] articulates trustworthiness as an end-to-end property spanning data collection, model development, deployment, and decommissioning. Achieving this standard for lattice-protected LLM systems requires solving three families of open problems whose solutions are mutually reinforcing: encrypted computation at LLM scale, automated cryptographic reasoning, and self-modifying encrypted models. None of these is achievable without the medium-term advances described above, and all three are prerequisites for deploying AI in settings where the HNDL threat model [20] or post-quantum regulatory mandates [21] apply.

10.3.1 *Universal encrypted computation*

The long-term vision is an LLM that is trained, fine-tuned, and deployed entirely within an encrypted computation pipeline, with post-quantum security guarantees from training data to inference output. This would require: (a) $< 10\times$ FHE overhead via hardware acceleration; (b) polynomial approximations accurate to 10^{-6} for all non-linear activations; and (c) post-quantum secure hardware attestation (via lattice-based signatures) for compute node integrity.

10.3.2 *Lattice-aware language model for cryptography*

A specialized LLM pre-trained on the mathematical literature of lattice cryptography, lattice reduction code (fplll, FLINT), and LWE estimator outputs could serve as a general-purpose assistant for cryptographers. Applications would include automated security proof verification, parameter optimization, and attack scenario simulation.

10.3.3 *Homomorphic model editing*

LoRA fine-tuning under FHE represents only the simplest form of encrypted model modification. Homomorphic model editing would allow arbitrary targeted modifications to a deployed LLM (e.g., inserting new knowledge, correcting factual errors) without decrypting the base model weights, using FHE to evaluate the editing objective on encrypted embeddings.

11. Conclusion

This survey has mapped the emerging intersection of lattice-based cryptography and large language models across three complementary frontiers. We have shown that LBC is not merely a post-quantum replacement for RSA and ECC, but a foundational building block for a new generation of privacy-preserving AI systems.

Frontier 1 (FHE-Protected LLM Training) establishes what encrypted training can and cannot deliver. It protects data from the compute provider, and it does not provide differential privacy: ciphertext error is not calibrated to sensitivity, does not survive decryption, and cannot substitute for clipping and calibrated noise on plaintext gradients. Beyond the computational overhead, which is improving through hardware acceleration, the binding constraints are the non-polynomial circuits enumerated in Section 4.3.2 and the per-step bootstrapping cost implied by depth that grows with network depth.

Frontier 2 (LLM-Guided Cryptanalysis) is a research proposal, and this review reaches a largely negative conclusion about it. No experiment demonstrates LLM-guided lattice reduction outperforming standard BKZ, and Proposition 5.1 shows that scheduling improvements alone cannot reduce core-SVP security regardless of how effective the scheduler becomes, because the limiting quantity is the minimum viable block size rather than the number of tours. The contribution of this section is consequently the falsifiable benchmark protocol of Section 5.6 and the explicit refutation of the tour-count reasoning that has appeared informally in this area. Practitioners should draw no parameter-selection implications from this frontier.

Frontier 3 (Lattice-Based FE for Inference) shows that inner-product and multi-client FE schemes built on LWE can enforce fine-grained, provably secure computation over transformer layers in decentralized settings. The theoretical foundations are solid; the engineering challenge of scaling to production LLM dimensions is the primary open problem. Across all three frontiers, the recurring themes are noise budget management, the security-utility tradeoff, the challenge of polynomial approximation for non-linear functions, and the transition from centralized to distributed trust models.

From an engineering standpoint, the three frontiers map to distinct deployment horizons, and none of them is close. Encrypted inference for small models is the most mature capability surveyed here at TRL 4, but it is an enabling result rather than one of the three frontiers. Of the frontiers themselves, lattice-FE reaches furthest: the ALS inner-product construction is at TRL 3 on the strength of a security proof and the toy implementation of Section 6.2, though FE-based transformer inference remains at TRL 1 because attention lies outside the inner-product class. FHE-protected fine-tuning follows at TRL 2, gated not by a missing circuit but by the throughput deficit quantified in Section 4.6; NTT hardware acceleration reducing per-operation overhead from $10^3\times$ to below $10\times$ is a precondition for moving it. LLM-guided lattice reduction is last at TRL 1: it is a stated goal with no supporting result, and Section 5.4 concludes that it presents no current basis for adjusting parameter selection. Practitioners should continue to follow the LWE estimator and NIST category guidance.

This review is intended to serve computational engineers, secure systems designers, and applied informaticians as a unified reference covering mathematical foundations, implementation complexity, TRL assessments, and hardware-software co-design requirements for AI systems that must remain private and quantum-resistant throughout the post-quantum transition.

Acknowledgments

The author thanks the open-source communities behind the fplll lattice reduction library, the OpenFHE framework, and the SCALE-MAMBA MPC system for providing the software foundations that make empirical research in this area accessible. The author also acknowledges the valuable suggestions from the peer reviewers, whose detailed technical criticism corrected three substantive errors in an earlier version of this manuscript.

Funding

This research received no external funding. No grant, contract, or other financial support from any public, commercial, or not-for-profit agency contributed to this work. As there was no funding party, no funding party participated in the study design, in the collection, analysis, or interpretation of data, in the writing of the manuscript, or in the decision to submit it for publication.

Declaration on the use of generative AI

In the preparation of this work, the author used Claude (Anthropic) to assist with text embellishment, literature organization, and structural refinement of the manuscript. After using this tool, the author reviewed and edited all content as needed, and takes full responsibility for the content of this publication. No AI tool was used to generate or alter any figure or image in this manuscript, and no reported measurement, benchmark, or citation was produced by an AI tool without verification against the primary source.

Data and code availability

The computational notebook that generates every figure identified as an original simulation in Table 2 accompanies this submission. It contains the complete source for each construction, parameter sweep, and plot, together with the fixed random seeds required to reproduce the reported values exactly. The environment in which the reported figures were produced is specified in Section 2.5. No other data were generated or analysed.

Conflict of interest

The author declares no conflict of interest. There are no financial interests or connections, direct or indirect, and no other situations that might raise the question of bias in the work reported or in the conclusions, implications, or opinions stated. The author has no commercial relationship with, and receives no support from, any vendor of the cryptographic libraries, hardware accelerators, trusted execution environments, or language models discussed in this review. Several works by the author are cited where directly relevant to the subject matter; these constitute a minority of the reference list and are identified as the author's own in the text.

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., "Language models are few-shot learners," in Advances in Neural Information Processing Systems 33 (NeurIPS 2020), 2020, pp. 1877-1901.

[2] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al., "Llama 2: Open foundation and fine-tuned chat models," *arXiv*, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2307.09288. [Accessed Aug. 1, 2026].

[3] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, et al., "Emergent abilities of large language models," *Transactions on Machine Learning Research*, pp. 1-30, 2022.

[4] National Institute of Standards and Technology, "Module-lattice-based key-encapsulation mechanism standard," Washington, DC, USA: U.S. Department of Commerce, Federal Information Processing Standards Publication 203, 2024.

[5] C. Peikert, "A decade of lattice cryptography," *Foundations and Trends in Theoretical Computer Science*, vol. 10, no. 4, pp. 283-424, 2016.

[6] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM*, vol. 56, no. 6, pp. 1-40, 2009.

[7] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in Advances in Cryptology-EUROCRYPT 2010, 2010, pp. 1-23.

[8] C. Gentry, *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, UWA, 2009.

[9] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," in Proceedings of the 3rd Innovations in Theoretical Computer Science (ITCS), 2012, pp. 309-325.

[10] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in Advances in Cryptology-ASIACRYPT 2017, 2017, pp. 409-437.

[11] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," in Theory of Cryptography Conference (TCC), 2011, pp. 253-273.

[12] D. Trivedi, A. Boudguiga, and N. Triandopoulos, "SigML: Supervised log anomaly with fully homomorphic encryption," in Cyber Security, Cryptology, and Machine Learning (CSCML 2023), 2023, pp. 372-388.

[13] D. Trivedi, A. Boudguiga, N. Kaaniche, and N. Triandopoulos, "SplitML: A unified privacy-preserving architecture for federated split-learning in heterogeneous environments," *Electronics*, vol. 15, no. 2, pp. 267, 2026.

[14] D. Micciancio and S. Goldwasser, *Complexity of Lattice Problems: A Cryptographic Perspective*. Boston, MA, USA: Kluwer Academic Publishers, 2002.

[15] M. R. Albrecht, R. Player, and S. Scott, "On the concrete hardness of learning with errors," *Journal of Mathematical Cryptology*, vol. 9, no. 3, pp. 169-203, 2015.

[16] MATZOV, "Report on the security of LWE: Improved dual lattice attack," Zenodo, 2022.

[17] J. Zhang, X. Cheng, L. Yang, J. Hu, X. Liu, and K. Chen, "SoK: Fully homomorphic encryption accelerators," *ACM Computing Surveys*, vol. 56, no. 12, pp. 1-32, 2024.

[18] National Institute of Standards and Technology, "FIPS 204: Module-lattice-based digital signature standard," USA: Federal Information Processing Standards Publication 204, 2024.

[19] National Institute of Standards and Technology, "FIPS 205: Stateless hash-based digital signature standard," USA: Federal Information Processing Standards Publication 205, 2024.

[20] National Cybersecurity Center of Excellence, "Migration to post-quantum cryptography: Preparation for considering the implementation and adoption of quantum safe cryptography," NIST Special Publication 1800-38 (Preliminary Draft), NIST NCCoE, 2023.

[21] National Institute of Standards and Technology, "Post-quantum cryptography." NIST, 2025. Available: https://www.nist.gov/pqc. [Accessed Aug. 1, 2026].

[22] M. Ajtai, "Generating hard instances of lattice problems," in Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC), 1996, pp. 99-108.

[23] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," Cryptology ePrint Archive, Report 2012/144, 2012.

[24] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "TFHE: Fast fully homomorphic encryption over the torus," *Journal of Cryptology*, vol. 33, no. 1, pp. 34-91, 2020.

[25] S. Agrawal, B. Libert, and D. Stehlé, "Fully secure functional encryption for inner products, from standard assumptions," in Advances in Cryptology-CRYPTO 2016, 2016, pp. 333-362.

[26] A. K. Lenstra, H. W. Lenstra, and L. Lovász, "Factoring polynomials with rational coefficients," *Mathematische Annalen*, vol. 261, no. 4, pp. 515-534, 1982.

[27] C. P. Schnorr and M. Euchner, “Lattice basis reduction: Improved practical algorithms and solving subset sum problems,” *Mathematical Programming*, vol. 66, pp. 181-199, 1994.
[28] National Institute of Standards and Technology, “NIST selects HQC as fifth algorithm for post-quantum encryption,” *NIST News*, 2025. [Online]. Available: https://www.comforte.com/fileadmin/Collateral/FS_TAMUNIO_Assure.pdf. [Accessed Aug. 1, 2026].
[29] National Institute of Standards and Technology, “Transition to post-quantum cryptography standards,” NIST, USA, NIST Internal Report 8547 (Initial Public Draft), 2024.
[30] National Institute of Standards and Technology, “Artificial intelligence risk management framework (AI RMF 1.0),” National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Trustworthy and Responsible AI NIST AI 100-1, Jan. 2023.
[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in Advances in Neural Information Processing Systems (NeurIPS), 2017, pp. 5998-6008.
[32] N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, Ú. Erlingsson, et al., “Extracting training data from large language models,” in 30th USENIX Security Symposium, 2021, pp. 2633-2650.
[33] N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramèr, and C. Zhang, “Quantifying memorization across neural language models,” *arXiv*, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2202.07646. [Accessed Aug. 1, 2026].
[34] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” in IEEE Symposium on Security and Privacy (S & P), 2017, pp. 3-18.
[35] M. Fredrikson, S. Jha, and T. Ristenpart, “Model inversion attacks that exploit confidence information and basic countermeasures,” in Proceedings of the 22nd ACM Conference on Computer and Communications Security (CCS), 2015, pp. 1322-1333.
[36] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” in International Conference on Learning Representations (ICLR), 2022.
[37] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, “Deep learning with differential privacy,” in Proceedings of the 23rd ACM Conference on Computer and Communications Security (CCS), 2016, pp. 308-318.
[38] D. Trivedi, N. Kaaniche, A. Boudguiga, and N. Triandopoulos, “chiku: Efficient probabilistic polynomial approximations library,” in Proceedings of the 21st International Conference on Security and Cryptography (SECRYPT 2024), 2024, pp. 634-641.
[39] J. Moon, D. Yoo, X. Jiang, and M. Kim, “THOR: Secure transformer inference with homomorphic encryption,” in Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS ’25), 2025, pp. 3765-3779.
[40] J. Zhang, X. Yang, L. He, K. Chen, W.-J. Lu, Y. Wang, X. Hou, J. Liu, K. Ren, and X. Yang, “Secure transformer inference made non-interactive,” in Proceedings of the 2025 Network and Distributed System Security Symposium (NDSS), 2025, pp. 136.
[41] B. Knott, S. Venkataraman, A. Hannun, S. Sengupta, M. Ibrahim, and L. van der Maaten, “CrypTen: Secure multi-party computation meets machine learning,” in Advances in Neural Information Processing Systems (NeurIPS), 2021, pp. 4961-4973.
[42] P. Mishra, R. Lehmkuhl, A. Srinivasan, W. Zheng, and R. A. Popa, “Delphi: A cryptographic inference service for neural networks,” in 29th USENIX Security Symposium, 2020, pp. 2505-2522.
[43] M. Hao, H. Li, H. Chen, P. Xing, G. Xu, and T. Zhang, “Iron: Private inference on transformers,” in Advances in Neural Information Processing Systems 35 (NeurIPS 2022), 2022, pp. 15718-15731.
[44] Y. Dong, W.-J. Lu, Y. Zheng, H. Wu, D. Zhao, J. Tan, Z. Huang, C. Hong, T. Wei, W.-G. Chen, et al., “Puma: Secure inference of llama-7b in five minutes,” *Security and Safety*, vol. 4, pp. 2025014, 2025.
[45] C. Dwork and A. Roth, *The Algorithmic Foundations of Differential Privacy*. Netherlands: Now Publishers, 2014.
[46] D. Yu, S. Naik, A. Backurs, S. Gopi, H. A. Inan, G. Kamath, J. Kulkarni, Y. T. Lee, A. Manoel, L. Wutschitz, et al., “Differentially private fine-tuning of language models,” in International Conference on Learning Representations (ICLR), 2022.
[47] C. L. Canonne, G. Kamath, and T. Steinke, “The discrete gaussian for differential privacy,” in Advances in Neural Information Processing Systems (NeurIPS), 2020, pp. 15676-15688.

[48] B. Reagen, W.-S. Choi, Y. Ko, V. T. Lee, H.-H. S. Lee, G.-Y. Wei, and D. Brooks, "Cheetah: Optimizing and accelerating homomorphic encryption for private inference," in IEEE International Symposium on High-Performance Computer Architecture (HPCA), 2021, pp. 26-39.

[49] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "GAZELLE: A low latency framework for secure neural network inference," in 27th USENIX Security Symposium, 2018, pp. 1651-1669.

[50] W.-J. Lu, Z. Huang, C. Hong, Y. Ma, and H. Qu, "PEGASUS: Bridging polynomial and non-polynomial evaluations in homomorphic encryption," in IEEE Symposium on Security and Privacy (S&P), 2021, pp. 1057-1073.

[51] Q. Pang, J. Zhu, H. Möllering, W. Zheng, and T. Schneider, "BOLT: Privacy-preserving, accurate and efficient inference for transformers," in 2024 IEEE Symposium on Security and Privacy (SP), 2024, pp. 4753-4771.

[52] H. Chen, I. Chillotti, and Y. Song, "Multi-key homomorphic encryption from TFHE," in Advances in Cryptology-ASIACRYPT 2019, 2019, pp. 446-472.

[53] D. Fiore, R. Gennaro, and V. Pastro, "Efficiently verifiable computation on encrypted data," in Proceedings of the 21st ACM Conference on Computer and Communications Security (CCS), 2014, pp. 844-855.

[54] Y. Aono, Y. Wang, T. Hayashi, and T. Takagi, "Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator," in Advances in Cryptology-EUROCRYPT 2016, 2016, pp. 789-781.

[55] B. Romera-Paredes, M. Barekatain, A. Novikov, M. Balog, M. P. Kumar, E. Dupont, F. J. R. Ruiz, J. S. Ellenberg, P. Wang, O. Fawzi, et al., "Mathematical discoveries from program search with large language models," *Nature*, vol. 625, no. 7995, pp. 468-475, 2024.

[56] M. R. Albrecht, S. Bai, P.-A. Fouque, P. Kirchner, D. Stehlé, and W. Wen, "Faster enumeration-based lattice reduction: Root Hermite factor k1/(2k) time kk/8+o(k)," in Advances in Cryptology-CRYPTO 2020, 2020, pp. 186-212.

[57] A. Becker, L. Ducas, N. Gama, and T. Laarhoven, "New directions in nearest neighbor searching with applications to lattice sieving," in Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2016, pp. 10-24.

[58] D. Selsam, M. Lamm, B. Bünz, P. Liang, L. de Moura, and D. L. Dill, "Learning a SAT solver from single-bit supervision," in International Conference on Learning Representations (ICLR), 2019.

[59] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in Advances in Cryptology-EUROCRYPT 1998, 1998, pp. 127-144.

[60] J. Chotard, E. Dufour-Sans, R. Gay, D. H. Phan, and D. Pointcheval, "Decentralized multi-client functional encryption for inner product," in Advances in Cryptology-ASIACRYPT 2018, 2018, pp. 703-732.

[61] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar, "Split learning for health: Distributed deep learning without sharing raw patient data," *arXiv*, 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1812.00564. [Accessed Aug. 1, 2026].

[62] J. M. Bermudo Mera, A. Karmakar, T. Marc, and A. Soleimanian, "Efficient lattice-based inner-product functional encryption," in Public-Key Cryptography-PKC 2022, 2022, pp. 163-193.

[63] M. Abdalla, F. Bourse, A. De Caro, and D. Pointcheval, "Simple functional encryption schemes for inner products," in Public-Key Cryptography-PKC 2015, 2015, pp. 733-751.

[64] S. Goldwasser, S. D. Gordon, V. Goyal, A. Jain, J. Katz, F.-H. Liu, A. Sahai, E. Shi, and H.-S. Zhou, "Multi-input functional encryption," in Advances in Cryptology-EUROCRYPT 2014, 2014, pp. 578-602.

[65] J. Zhu, H. Yin, P. Deng, A. Almeida, and S. Zhou, "Confidential computing on NVIDIA Hopper GPUs: A performance benchmark study," *arXiv*, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2409.03992. [Accessed Aug. 1, 2026].

[66] M. Misono, D. Stavrakakis, N. Santos, and P. Bhatotia, "Confidential VMs explained: An empirical analysis of AMD SEV-SNP and Intel TDX," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 8, no. 3, pp. 1-42, 2024.

[67] S. Liu, Z. Wang, Y. Chen, and Q. Lei, "Data reconstruction attacks and defenses: A systematic evaluation," *arXiv*, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2402.09478. [Accessed Aug. 1, 2026].

[68] E. Erdoğan, A. Küpçü, and A. E. Çiçek, "UnSplit: Data-oblivious model inversion, model stealing, and label inference attacks against split learning," in Proceedings of the 21st Workshop on Privacy in the Electronic Society (WPES), 2022, pp. 115-124.

[69] N. Samardzic, A. Feldmann, A. Krastev, N. Manohar, N. Genise, S. Devadas, K. Eldefrawy, C. Peikert, and D. Sanchez, "CraterLake: A hardware accelerator for efficient unbounded computation on encrypted data," in Proceedings of the 49th Annual International Symposium on Computer Architecture (ISCA), 2022, pp. 173-187.

[70] A. Al Badawi, J. Bates, F. Bergamaschi, D. B. Cousins, S. Erabelli, N. Genise, S. Halevi, H. Hunt, A. Kim, Y. Lee, et al., "OpenFHE: Open-source fully homomorphic encryption library," in Proceedings of the 10th Workshop on Encrypted Computing and Applied Homomorphic Cryptography (WAHC), 2022, pp. 53-63.

[71] C. Agulló-Domingo, Ó. Vera-López, S. Guzelhan, L. Daksha, A. El Jerari, K. Shivdikar, R. Agrawal, D. Kaeli, A. Joshi, and J. L. Abellán, "FIDESlib: A fully-fledged open-source FHE library for efficient CKKS on GPUs," in 2025 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2025, pp. 1-3.

[72] I. Mironov, O. Pandey, O. Reingold, and S. Vadhan, "Computational differential privacy," in Advances in Cryptology-CRYPTO 2009, 2009, pp. 126-142.