VEILMOE: EFFICIENT SECURE MOE INFERENCE WITH FAITHFUL SPARSE EXECUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Secure mixture-of-experts (MoE) inference extends privacy-preserving Transformer inference beyond dense computation to sparse expert activation. However, existing secure MoE frameworks impose routing or expert-capacity restrictions that force single-expert selection, dropped assignments, or altered selection semantics that deviate from native plaintext MoE execution. They also overlook the differences in weight visibility across open-source and closed-source deployments, incurring unnecessary cryptographic overhead. In this work, we present VeilMoE, an efficient two-party privacy-preserving inference framework based on function secret sharing (FSS) that protects input-dependent routing while preserving exact, drop-free Top-k semantics and native expert sparsity in both settings. In the closed-source setting, VeilMoE-C secretly permutes and masks the expert bank in the offline phase and reveals only selected anonymized slots, with the resulting residual linkability formally characterized and empirically quantified. In the open-source setting, VeilMoE-O employs a route-independent access pattern to conceal the selected experts while avoiding cryptographic evaluation of the full expert bank. Extensive experiments show that VeilMoE reduces end-to-end online communication and latency by up to $7.2\times$ and $2.9\times$, respectively, compared with the state-of-the-art FSS-based dense secure-inference baseline.

1 INTRODUCTION

Privacy-preserving Transformer inference lets users query a model without revealing their input or the owner's private weights to inference servers. Existing systems use homomorphic encryption (HE), secure two-party computation (2PC), and function secret sharing (FSS) to evaluate dense Transformer layers (Hao et al., 2022; Pang et al., 2024; Gupta et al., 2024; Lu et al., 2025). Modern language models increasingly use mixture-of-experts (MoE) layers, which route each token to k of N expert feed-forward networks (FFNs) (Jiang et al., 2024; Dai et al., 2024; Muennighoff et al., 2025). This input-dependent routing creates an additional privacy surface: expert identities and recurring co-activations can convey information about queries (Ding et al., 2025; Nuriyev & Kulp, 2026). Our objective is to protect MoE inference while retaining the expert contributions prescribed by the original model, rather than changing its routing policy to simplify security.

Native MoE execution computes k selected FFNs (Figure 1). A route-independent evaluation of all N experts can preserve the result but pays for dense cryptographic computation. Opening native expert indices preserves sparse access but exposes routing. Existing secure MoE systems take different approaches: SecMoE targets single-expert selection, whereas CryptoMoE bounds expert capacity and discards overflow assignments (Shen et al., 2026; Zhou et al., 2025). Single-expert selection does not reproduce a checkpoint with native $k > 1$; capacity-limited execution can omit a selected expert's contribution. We ask whether secure inference can instead keep the native activation count, avoid capacity drops, and cryptographically evaluate only selected FFNs.

The difficulty extends beyond identifying the largest scores. A protected selection must drive expert access, align each expert with its gate coefficient, and feed the selected computation into subsequent masked layers. Public weights permit local linear operations on secret selectors. Private weights instead permit direct sparse dispatch through secret relabeling when anonymous membership is an acceptable release. Both target the same MoE computation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We introduce **VeilMoE**, which derives masked ranks once and uses them to drive complete expert evaluation and aggregation (Figure 2). VeilMoE-C jointly permutes the private router and expert bank, opens only shuffled membership, and evaluates the selected masked experts. VeilMoE-O keeps selection shared and gathers public expert matrices in a fixed full-bank scan. VeilMoE-CG provides private-weight gather when membership must also remain hidden. The gather variants scan all experts locally but cryptographically evaluate only k FFNs. None imposes a capacity-based drop rule. Exactness refers to the specified fixed-point execution and stable ordering; equivalence to a checkpoint's native selected set additionally requires an unchanged Top-k boundary, as detailed in Section 2.1.

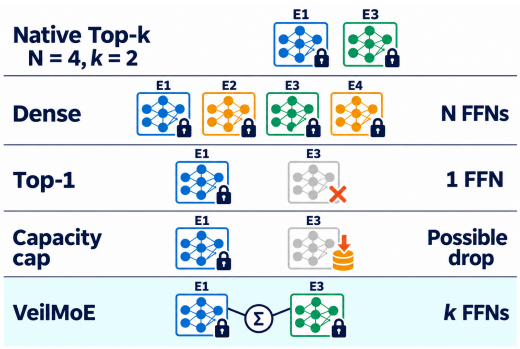

Figure 1: A token within a batch, with $N = 4, k = 2$. Other tokens can exhaust an expert's capacity. VeilMoE retains both selected contributions without evaluating all four FFNs.

Our contributions are summarized as follows: (1)We design a unified FSS-based 2PC framework that derives masked stable router ranks once, then adapts them either through a membership threshold for private-weight shuffled dispatch or through cumulative-threshold differencing for public-weight oblivious gather. Both realize exact, drop-free Top-k semantics while cryptographically evaluating only k FFN payloads. (2)We formalize the routing observation released by each variant and use controlled attacks to separate anonymous co-activation linkability and semantic transfer from recovery of native expert identities. (3)Compared with the state-of-the-art (SOTA) FSS-based dense secure-inference baseline (Gupta et al., 2024), VeilMoE cuts end-to-end online communication by up to $7.2\times$ and end-to-end online latency by up to $2.9\times$.

Related work. Dense secure Transformer systems optimize matrix products, nonlinearities, and HE–MPC boundaries (Gupta et al., 2022; 2024; Jawalkar et al., 2024; Li et al., 2024; Kei & Chow, 2025). These optimizations complement, rather than replace, MoE-specific expert selection. SecMoE (Shen et al., 2026) and CryptoMoE (Zhou et al., 2025) directly address secure MoE but use different routing and deployment contracts. We compare their routing restrictions on identical checkpoints; this does not reproduce their cryptographic runtimes. Token-pruning and Transformer approximation systems similarly expose a computation–accuracy tradeoff (Zhang et al., 2025; Zhu et al., 2026). VeilMoE instead retains every selected expert contribution. We build on CmpAgg ranking (Agarwal et al., 2024) and FSS inference primitives, and contribute their integration with protected expert access. Routing attacks motivate an explicit analysis of the released stream rather than treating expert relabeling as unlinkability (Yona et al., 2024; Hayes et al., 2024).

2 Preliminary

2.1 MoE computation and fidelity

A Transformer alternates token-mixing attention with position-wise FFNs (Vaswani et al., 2017). An MoE replaces the latter with a router and N experts. For a row-vector layer input x, let $g = xW_r$ be its router scores, S its selected k experts, and $w_e(g, S)$ its checkpoint-defined gate coefficient, including any selected-weight renormalization. For SwiGLU experts (Dai et al., 2024; Muennighoff et al., 2025),

$$\mathsf{FFN}_e(x) = \big(\mathrm{SiLU}(xW_e^{\mathrm{gate}}) \odot xW_e^{\mathrm{up}}\big)W_e^{\mathrm{down}}. \tag{1}$$

The routed output is $y = \sum_{e\in S} w_e(g, S)\mathsf{FFN}_e(x)$. Always-on shared experts, when present, add their prescribed contributions outside the routed set. *Drop-free* means that all k selected routed experts contribute, without capacity-based omission or rerouting.

A *stable order* sorts scores with an explicit deterministic secondary key; the stable rank ρ_e counts experts below e in that order, so the largest has rank $N - 1$. Our measured implementation breaks equal scores by the smaller current slot index. The Top-k boundary is *tie-free* when the kth and

$(k+1)$st scores differ. Under this condition a permutation does not change the selected set. At a boundary tie, VeilMoE-C and VeilMoE-CG can select a different tied subset from a native-index rule. We do not identify stable cardinality with unconditional native tie equivalence. Moreover, quantization can change a boundary before secure selection. We distinguish *routing fidelity* (selection and weighting), *numerical correctness* (the specified fixed-point program), and *task accuracy* (the downstream score).

2.2 Parties and threat model

A user U and model owner O outsource computation to non-colluding servers $\mathsf{P}_0, \mathsf{P}_1$. A static probabilistic polynomial-time adversary corrupts at most one server. Semi-honest means it executes the protocol but inspects its complete state and transcript. A trusted preprocessing dealer D, outside the corruption scope, supplies masks, FSS keys and correlations, then leaves before online execution (Gupta et al., 2022; 2024). The user sends a masked post-embedding input and receives the masked output; it need not participate in intermediate online steps. This differs from an interactive HE client–server deployment. No complete input or output mask is given to either server.

The target is confidentiality of inputs, intermediate values, private weights, and outputs, subject to explicit public metadata and routing leakage. Architecture, batch shape and prescribed message sizes are public. VeilMoE-C releases the ordered stream of shuffled membership sets; it does not promise that repeated queries are unlinkable. VeilMoE-O and VeilMoE-CG hide that modeled access trace. Collusion, malicious deviation and arbitrary hardware side channels are not covered. The current private-gather key generator consumes private weight values inside the trusted preprocessing boundary; it is not a dealer-blind instantiation. Appendix A details deployment and failure handling.

2.3 Notation and cryptographic interfaces

Let $[n] = \{0, \ldots, n-1\}$. W_e^p is expert e's matrix for projection $p \in \{\mathrm{gate}, \mathrm{up}, \mathrm{down}\}$; W without indices is the entire bank. x, y are layer values, whereas x_{emb} and y_{out} are model-boundary values. A uniform mask r_v gives a public masked value $\hat{v} = v + r_v$. In particular, $r_{x_{\mathrm{emb}}}$ masks the user's embedded input and $r_{y_{\mathrm{out}}}$ masks the final output; the dealer sends both to U. The owner receives complete private-weight masks and publishes only masked weights. The superscript $\prime$ denotes a permuted bank, $\widetilde{W}_j^p$ a gathered matrix, and ρ always denotes rank, not a mask.

Arithmetic shares satisfy $v = \langle v \rangle_0 + \langle v \rangle_1$ modulo the working ring. An opening exchanges the two shares and reconstructs their sum; Boolean shares $[\![c]\!]_{2,b}$ instead reconstruct by XOR. The implementation uses 50-bit model values and 64-bit rank/selector working arithmetic; selectors are reduced modulo 2^{50} at the model interface. Machine-word storage does not make the two ring widths identical. Boolean-to-arithmetic conversion preserves a predicate in shares. Beaver triples (Beaver, 1992) provide shared products: with $c = ab$, opening only $d = u - a$ and $e = v - b$ permits local shares of $uv = c + db + ea + de$.

FSS (Boyle et al., 2015; 2016) represents a function f by two keys: $o_b = \mathrm{Eval}(b, \kappa_b, z)$ satisfies $o_0 + o_1 = f(z)$ on public input z, while one key hides the function. A comparison function returns β for $z < \alpha$ and zero otherwise. Preprocessing encodes masked functions $z \mapsto f(z - r_{\mathrm{in}}) + r_{\mathrm{out}}$ in the keys. FSS keys are not reconstructible shares of a mask; arithmetic mask shares and product correlations are separate objects when an operation needs them. Secure linear, softmax, SiLU and multiplication interfaces follow FSS-based inference (Gupta et al., 2022; 2024; Jawalkar et al., 2024).

3 VeilMoE Framework

VeilMoE connects protected ranks to expert computation (Figure 2). Routing, nonlinear evaluation and weighted aggregation are shared; expert access changes with weight visibility and the permitted membership release. Table 1 isolates this design choice from dense execution and restricted routing policies.

For T tokens, $C_{\mathrm{cap}} = \lceil cTk/N \rceil$ expresses per-expert capacity, with public capacity multiplier c. SecMoE and CryptoMoE use interactive HE/MPC client–server protocols (Shen et al., 2026;

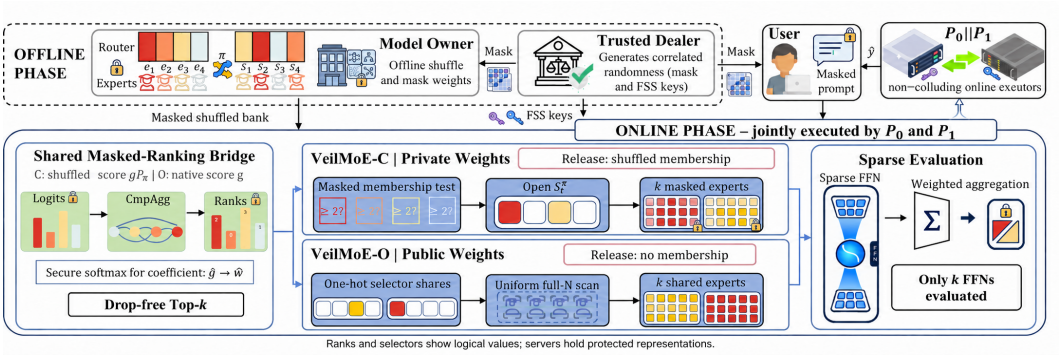

Figure 2: VeilMoE's routing-to-execution workflow. Private-bank dispatch (upper) or public-bank gather (lower) feeds k secure FFNs and weighted aggregation; VeilMoE-CG extends gather to private shares. "Masked prompt" means masked post-embedding input. Ranks and selectors are logical examples; locks denote masking or sharing, not HE.

Table 1: Routing and logical routed-FFN evaluations per layer for T tokens. Counts include padding, exclude shared experts and non-FFN stages, and do not predict cross-system runtime.

Framework	Routing policy	FFNs	Access; membership release
Dense FSS reference	Top-k, drop-free	TN	All experts; none
SecMoE	Top-1	T	HE bank gather; none
CryptoMoE	Capacity-limited Top-k	NC_{cap}	Padded expert batches; none
VeilMoE-C	Top-k, drop-free	Tk	k blocks/token; S^π
VeilMoE-O	Top-k, drop-free	Tk	Public-bank scan; none
VeilMoE-CG	Top-k, drop-free	Tk	Private-share-bank scan; none

Zhou et al., 2025); our dense reference and VeilMoE use two non-colluding servers with trusted preprocessing. Replacing TN with Tk removes unselected FFN evaluations, not all bank-wide work. Drop-free execution retains the tie and numerical qualifications of Section 2.1.

3.1 Shared Masked Routing

Aligned router and expert bank. In VeilMoE-C, the model owner O samples a layer permutation $\pi : [N] \to [N]$, with $\pi(e)$ the slot occupied by native expert e. Define $(P_\pi)_{e,s} = \mathbf{1}\{s = \pi(e)\}$, so $(gP_\pi)_s = g_{\pi^{-1}(s)}$. After receiving weight masks from the trusted dealer D, the owner publishes the masked router $\hat{W}'_r = W_r P_\pi + r_{W_r}$ and expert matrices $\hat{W}'^p_s = W^p_{\pi^{-1}(s)} + r^p_{W,s}$ for each projection p. All three matrices of one expert move together. Neither server knows the permutation or a complete weight mask. VeilMoE-CG uses the same masked private router with an aligned share bank; VeilMoE-O uses public weights in native order.

From masked values to protected ranks. Both servers start with $\hat{x} = x + r_x$. For an activation v entering a linear stage with matrix L, let $a_{\text{raw}} = vL$. Our interface Π_{Lin} takes public masked operands and each server's private preprocessing state, and returns $\hat{a} = \text{Trunc}_f(a_{\text{raw}}) + r_a$, where f is the fractional precision. Its raw private-weight product expands $(\hat{v} - r_v)(\hat{L} - r_L)$ using preprocessed arithmetic correlations, following Gupta et al. (2022; 2024); Appendix A.2 gives the local formula. Applying this interface to the router gives $\hat{g} = \bar{g} + r_g$, where $\bar{g} = gP_\pi$ for private weights and $\bar{g} = g$ for public weights. Secure softmax consumes the same masked scores and produces masked gate coefficients $\hat{w} = w + r_w$. The two measured checkpoints use full-bank softmax coefficients without selected-set renormalization; a different checkpoint's gate convention must be implemented explicitly.

We use CmpAgg's compare-and-aggregate ranking (Agarwal et al., 2024), not a new sorting primitive. Initialize all arithmetic rank shares to zero. For each pair $a < b$, the servers securely obtain

shares of $\chi_{ab} = \mathbf{1}\{\bar{g}_a \geq \bar{g}_b\}$ and add $\langle \chi_{ab} \rangle$ to $\langle \rho_a \rangle$ and $1 - \langle \chi_{ab} \rangle$ to $\langle \rho_b \rangle$. Masked Boolean comparisons are converted to arithmetic shares without opening their unmasked predicates. Each rank counts the experts it outranks; rank $N-1$ is largest. Equal scores favor the smaller current slot, yielding a strict total order and distinct ranks $0, \ldots, N-1$. The servers add preprocessed rank-mask contributions and open only $\hat{\rho} = \rho + r_\rho$.

This gives exactly k selected slots, subject to the tie convention in Section 2.1. The oblivious path accumulates ranks and evaluates thresholds in 64 bits; reducing its $0/1$ selector shares locally to the 50-bit model ring preserves their sum without opening them.

3.2 VeilMoE-C: Shuffled Sparse Dispatch

A membership threshold on each masked rank returns Boolean shares of $m_s = \mathbf{1}\{\rho_s \geq N - k\}$. Both servers exchange these shares and recover $S^\pi = \{s : m_s = 1\}$, the authorized shuffled membership release. They then access the same k masked expert blocks. Native expert labels are not transmitted, but repeated slot memberships remain observable.

For each selected block, the servers evaluate the gate and up projections on $\hat{x}$, followed by secure SiLU and element-wise multiplication. The resulting masked SwiGLU activation is the input to the down projection; it is not another use of the original x. Each projection produces a fresh masked output. Secure multiplication applies the corresponding protected gate coefficient, and addition forms the masked layer output $\hat{y}$. No capacity buffer removes an assignment. Protocol 1 names the actors and the representation exchanged at each boundary.

Protocol 1 Shared routing and variant-specific expert access.

Input: Servers hold $\hat{x}$; weights follow Table 1; (N, k) is public.
Output: $\hat{y}$ at both servers; VeilMoE-C additionally opens S^π.

1: **Offline,** D**:** Deliver fresh routing keys and stage-specific correlations to each server; deliver complete boundary masks only to data owners.
2: **Offline,** O**:** Publish aligned masked private weights (VeilMoE-C) or provision the private share bank (VeilMoE-CG).
3: **Servers:** Securely compute masked scores $\hat{g}$ and coefficients $\hat{w}$.
4: **Servers:** Evaluate FSS comparisons and jointly open masked Boolean outputs; locally apply B2A and accumulate $\langle \rho \rangle_b$.
5: **Servers:** Add rank masks and reconstruct only $\hat{\rho}$.
6: **VeilMoE-C, servers:** Evaluate membership keys, exchange Boolean outputs, recover S^π, and fetch its k masked expert blocks.
7: **Gather variants:** Each server differences threshold outputs to get $\langle q_j \rangle_b$ and gathers public matrices locally. For private matrices, servers first open $\Delta = Q - A$, then apply Eq. (4) locally.
8: **Gather variants, servers:** Gather gate coefficients by Beaver inner products and use Eq. (3) for each masked projection.
9: **Servers:** Evaluate the selected k FFNs, securely apply aligned gate coefficients and sum. Reconstruct only freshly masked stage outputs and $\hat{y}$.

3.3 VeilMoE-O: Oblivious Expert Gather

Public weights make opened expert slots identifiable. VeilMoE-O therefore keeps selection shared and uses a fixed-order scan of the public bank. The key step is to turn cumulative rank thresholds into one selector per selected expert, then carry that selector through masked expert computation.

From cumulative thresholds to one-hot selectors. For $\lambda \in [k]$, the servers locally evaluate arithmetic-output FSS keys for $\iota_{\lambda,e} = \mathbf{1}\{\rho_e \geq N - k + \lambda\}$. With $\iota_{k,e} = 0$, define $q_{j,e} = \iota_{k-1-j,e} - \iota_{k-j,e}$ for $j \in [k]$ and gather

$$\widetilde{W}_j^p = \sum_{e \in [N]} q_{j,e} W_e^{\mathrm{pub},p}. \tag{2}$$

Distinct ranks make q_j one-hot, selecting the $(j+1)$-st largest expert (Figure 3). Unlike the DPF equality extraction of Agarwal et al. (2024, Sec. 4.2), our construction uses kN cumulative comparisons on the same comparison backend.

Each server weights every public expert matrix by its own selector share and sums locally. The two resulting shares already sum to the selected matrix in Eq. (2); gathering itself needs no reconstruction. The scan touches every expert even though the logical selector is one-hot. Neither server can identify its nonzero position from its random shares.

The selected gate coefficient must follow the same selector. The servers first subtract their preprocessed mask contributions from the masked coefficients, assigning the public term to one party, then use a Beaver inner product to obtain $\langle \widetilde{w}_j \rangle$. This keeps expert weights and gate weights aligned without opening either selection.

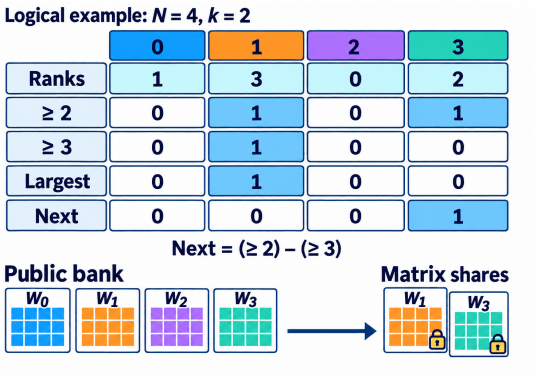

Figure 3: Threshold differences isolate one expert. Binary entries are logical values; servers hold random shares and never open the selected matrix.

Using a shared matrix with a masked activation. The next stage must compute $v\widetilde{W}_j^p$, not $\hat{v}\widetilde{W}_j^p = v\widetilde{W}_j^p + r_v\widetilde{W}_j^p$. For this stage the dealer shares $T_e^p = r_v W_e^{\mathrm{pub},p}$ and a correlation $\tau_j^p = \sum_e r_{q,j,e} T_e^p$, where $r_{q,j}$ is a fresh selector mask. The servers use the corresponding arithmetic correlation shares to open $\delta_j = q_j - r_{q,j}$, then obtain

$$\langle r_v \widetilde{W}_j^p \rangle = \langle \tau_j^p \rangle + \sum_e \delta_{j,e} \langle T_e^p \rangle. \tag{3}$$

Each party performs its product and correction locally; the exchange combines only freshly masked results. Figure 5 in Appendix A.2 illustrates these local and exchanged values. They subtract this contribution locally, add a fresh output mask, and reconstruct only the masked projection. Thus the selector opening supports mask correction; it does not expose q_j or reconstruct the selected matrix. Stage- and rank-dependent activation masks are implicit in Eq. (3); different hidden activations use their own correlations.

3.4 VeilMoE-CG: Private Weights Without Membership Release

VeilMoE-CG provides the same fixed-access policy when expert weights are private. The measured one-pass implementation uses a persistent arithmetic share bank $\langle W'^p \rangle$, where W'^p stacks the privately permuted experts. It does not publish or reconstruct this bank. Let Q stack the k one-hot selectors and let A be a fresh uniformly random matrix of the same shape. Preprocessing supplies $\langle A \rangle$ and $\langle AW'^p \rangle$. After opening $\Delta = Q - A$, each server computes

$$\langle QW'^p \rangle_b = \Delta \langle W'^p \rangle_b + \langle AW'^p \rangle_b. \tag{4}$$

This fixed scan yields the selected private matrices in shares. Activation correction uses $T_e^p = r_v W_e'^p$ in Eq. (3); FFN evaluation follows VeilMoE-O. The current CG preprocessing implementation consumes plaintext private weights to generate these correlations. It therefore trusts the preprocessing service with the model, in addition to trusting it with masks; it does not implement model-blind correlation generation.

All variants cryptographically evaluate k FFNs, but their total work is different. Shared pairwise ranking costs $O(N^2)$ comparisons. VeilMoE-C then fetches k expert blocks, whereas VeilMoE-O and VeilMoE-CG perform a full-bank selection scan. For D_W entries per expert, that scan costs $O(kND_W)$ local arithmetic; small communication does not remove this memory and compute cost. CG trades additional preprocessing and scan work for suppression of the membership channel. It is a stronger access policy, not an equally protected speed optimization of VeilMoE-C.

4 CORRECTNESS AND ROUTING PRIVACY

Correctness. Pairwise ranking defines a total order, so the membership threshold selects exactly k slots and adjacent cumulative thresholds yield disjoint one-hot selectors. Joint permutation preserves expert–coefficient alignment; Eqs. (3) and (4) preserve the selected linear products. Composition with the specified secure FFN and aggregation primitives therefore evaluates the reference fixed-point MoE with the protocol's stated tie rule. Native selected-set equivalence requires a tie-free boundary, and numerical equality is to that reference circuit, not an unrestricted floating-point execution.

What is protected. Let pub contain the architecture, public checkpoint when applicable, tensor shapes, token and query boundaries, and prescribed execution schedule. For token t, layer h, and permutation epoch η, define

$$\mathcal{L}_C = \left(S_{t,h}^{\pi_{h,\eta}}\right)_{t,h,\eta}. \tag{5}$$

This is the complete ordered membership stream, not merely its aggregate frequencies. The gather variants instead have $\mathcal{L}_{\mathrm{CG}} = \mathcal{L}_O = \varnothing$.

Theorem 1 (Security under the stated preprocessing model)**.** *Assume the masked-linear, FSS, B2A, nonlinear, and Beaver interfaces are secure against one static semi-honest server with the joint correlation interface in Appendix A.3 and fresh material. For a fixed public execution schedule, each variant's server view is computationally simulatable from* pub *and its leakage in Eq.* (5)*.*

The proof composes the primitive simulators while preserving their shared wire and preprocessing dependencies. It inserts VeilMoE-C's authorized memberships or the gather variants' fixed scans. It does not assume that correlated keys and messages can be sampled independently. Appendix A gives the assumptions and composition argument.

Interpretation. For executed native membership indicators $z_{t,h}$, the disclosed indicators are $z'_{t,h} = P_{\pi_h}^\top z_{t,h}$. Thus their co-activation matrices obey $G'_h = P_{\pi_h}^\top G_h P_{\pi_h}$, where $G_h = \sum_t z_{t,h} z_{t,h}^\top$. This is one statistic derivable from $\mathcal{L}_C$, not an upper bound on its information. Repeated memberships can support linkage and clustering even without native expert names. The theorem promises no additional protocol leakage; it does not make the authorized stream harmless. VeilMoE-CG and VeilMoE-O remove this membership channel when it is unacceptable. Their empty routing leakage concerns the modeled transcript and access schedule, not malicious deviation, server collusion, or unmodeled hardware side channels.

5 EXPERIMENTS

We evaluate routing fidelity, policy accuracy, resource costs and residual leakage separately, using the evidence appropriate to each question.

Setup and measurement scope. We use OLMoE and DeepSeekMoE checkpoints with $N = 64$ and native $k = 8$ and 6, respectively. DeepSeekMoE's initial dense layer and always-on shared experts are retained outside routed-expert selection. Each server has an NVIDIA A100 GPU; the main network comparison uses LAN and a 400-Mbit/s link with 10-ms RTT. The main tables report arithmetic means, normally over three independent runs, with five runs for designated unstable cells. Sample standard deviations (SD) and repetition counts are given in Appendix C.3. Appendix C specifies hardware, precision, datasets and network accounting.

The timing harness executes an architecture-sized attention–MoE stack with zero attention/expert payload weights and a synthetic router. It measures cryptographic cost, not trained-checkpoint generation. Dense-SIGMA is a private-weight reference using the same stack, backend and native k; it evaluates all N experts before masking unselected contributions. Nonzero-weight tests check correctness separately.

Routing fidelity and task accuracy. The nonzero-weight two-party check verifies all 32 outputs of a small VeilMoE-CG instance with maximum absolute error 6.47×10^{-4}. A real-weight, 16-token OLMoE check has complete selected-set agreement for its captured layer inputs. Appendix B reports

numerical errors and the plaintext-backbone splice used for that validation; it does not establish accumulated encrypted execution or long-generation stability.

We additionally audit 25,040 OLMoE and 42,984 DeepSeekMoE layer–token routing decisions on 20 predetermined documents per checkpoint. With 12 fractional bits, boundary ties occur in 0.112% and 0.133% of decisions; a seeded permutation changes the selected set in 0.052% and 0.070%. Increasing fractional precision to 14 reduces quantization-induced set changes against the same-input widened reference from 0.994% to 0.172% for OLMoE and from 0.584% to 0.102% for DeepSeekMoE. This confirms that the tie-free qualification matters even at these precisions.

Table 2 instead measures the *task cost of routing restrictions* using quantized plaintext MoE blocks in the native backbone. Native-k drop-free execution is VeilMoE's reference policy, not a full encrypted benchmark run. Top-1 uses one expert with unit gate weight on the same checkpoint; it is not SecMoE's separately trained model. The capacity policies emulate CryptoMoE's overflow rule at $c = 1$ and $c = 2$. All other task settings are matched. The eight-task general score is macro raw accuracy; MathQA and PIQA use length-normalized likelihood accuracy. PIQA is also in the general suite, so the rows are not independent replications.

Table 2: Task accuracy (%) under matched routing policies. VeilMoE's reference keeps native k; Top-1 and capacity limits alter the same checkpoint.

Model	Workload	VeilMoE ref.	Top-1	Capacity $c = 1$	Capacity $c = 2$
OLMoE	General	60.26	33.83	57.30	59.81
	MathQA	29.65	18.06	25.23	27.71
	PIQA	80.90	49.40	74.16	79.00
DeepSeekMoE	General	59.67	33.33	58.18	59.65
	MathQA	31.29	20.47	27.44	30.15
	PIQA	80.20	50.33	78.35	80.03

At $c = 1$, paired 95% bootstrap loss intervals exclude zero for all six rows; the largest loss is 6.75 points on OLMoE PIQA. At $c = 2$, DeepSeekMoE's general and PIQA losses are not distinguishable from zero, but MathQA remains sensitive for both checkpoints. Preserving contributions avoids this task-dependent policy tradeoff. Appendix C.2 gives drop rates and intervals, separately from fixed-point numerical error.

Communication and latency. Table 3 establishes savings against matched dense FSS execution. The largest communication reduction is $7.2\times$ for DeepSeekMoE with VeilMoE-C; the corresponding ratio of mean LAN times is $2.9\times$. Public-bank gathering retains a similar communication advantage, but costs more local work than shuffled dispatch. Ratios match architecture, token count and network, not weight-visibility and leakage contracts; all exclude preprocessing. LAN measurements have substantial variability, especially for DeepSeekMoE, so the ratios of means should not be read as guaranteed request-level speedups.

Table 3: Four-token architecture-sized stack cost: mean latency and per-party traffic. WAN has 10-ms RTT; SD and run counts are in Appendix C.3.

Model	Method	LAN (s)	WAN (s)	Traffic (MiB)
OLMoE	Dense-SIGMA	16.17	38.21	1248.40
	VeilMoE-C	6.01	18.37	190.37
	VeilMoE-O	8.29	21.60	191.95
DeepSeekMoE	Dense-SIGMA	36.98	81.10	2520.48
	VeilMoE-C	12.57	38.38	351.87
	VeilMoE-O	20.17	47.39	353.87

The same distinction holds for DeepSeekMoE: VeilMoE-CG's four-token mean is 37.47 seconds, similar to the dense baseline's mean despite about seven times fewer online bytes.

The two panels in Figure 4 contrast bytes and elapsed time. Batched nonlinear stages mostly grow wider, rather than adding sequential stages: message sizes increase with tokens, while rounds change little. At high RTT, shared synchronization cost can narrow latency gaps despite large byte savings. Appendix C.3 reports 20-, 80- and 160-ms profiles; the latter two are stress conditions. This token sweep does not establish expert-count scalability: ranking is quadratic in N, and fixed-k gathers scan a bank whose size grows with N.
VeilMoE-CG illustrates the remaining local cost. Its private share-bank scan keeps online bytes close to sparse dispatch but requires more memory traffic and fresh correlations. The same small message can therefore accompany very different local work. This distinction explains why byte savings should not be interpreted as equal latency gains. CG offers stronger access privacy, not the fastest private-weight execution; further scan parallelization is an optimization opportunity, not a measured benefit.

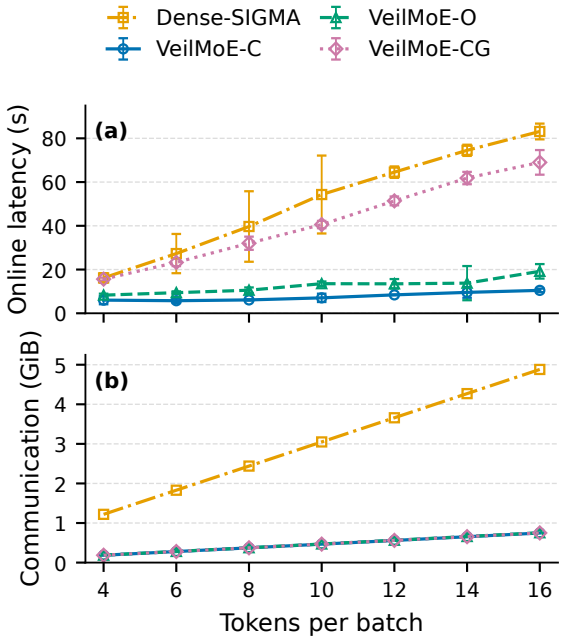

Figure 4: OLMoE token scaling on LAN: (a) mean latency with sample SD bars; (b) per-party bytes. Cells use three or five runs. Four-token anchors match Table 3; lines guide the eye.

Fresh material falls by up to $4.1\times$ with VeilMoE-C and VeilMoE-O, but private gather needs more than dense execution (Table 4). Fresh material is consumed per call; model-bank state is reused. Public weights remove private-bank storage, not fresh correlations. Appendix C.4 gives generation time and bank size.

Table 4: Fresh material per party (GiB), four tokens.

Method	OLMoE	DeepSeekMoE
Dense-SIGMA	24.10	52.65
VeilMoE-C	5.89	14.55
VeilMoE-O	5.93	14.63
VeilMoE-CG	29.95	56.40

No matched execution of SecMoE or CryptoMoE is included: stacks, online client duties, preprocessing and routing differ. Policy emulation is not a runtime comparison. Short-stack timings also do not establish long prefill, KV-cache decoding throughput or multi-user performance.

Routing leakage. Under the tested unrelated public-proxy alignment, native-expert recovery stays near $1/64$. However, clustering VeilMoE-C's actual slot stream gives adjusted Rand indices of 0.717 for OLMoE and 0.522 for DeepSeekMoE, with permuted-label controls near zero. Hidden expert names therefore do not imply hidden document structure. Stronger auxiliary checkpoints can also improve identity recovery. When such structure is sensitive, the fixed-access gather variants remove the membership channel by construction. Appendix C.6 details auxiliary-model controls, repeated-query linkage and cross-layer tests of the authorized stream.

6 Conclusion

VeilMoE connects protected Top-k routing to complete evaluation of the selected MoE experts. Shared masked ranking supports shuffled private-bank dispatch, public-bank oblivious gather, and a private-gather extension with stronger access protection. Architecture-sized measurements show up to $7.2\times$ lower online communication and a $2.9\times$ ratio of mean LAN times over dense FSS execution. The benefit is retaining all selected expert contributions while making the access and preprocessing tradeoff explicit.

AI USE STATEMENT

Generative AI tools assisted with manuscript editing, literature discovery, and internal manuscript critique. Numerical results are derived from recorded executions and evaluation files, not generated by an image model or language model.

ETHICS STATEMENT

This work studies confidentiality in outsourced inference using public model checkpoints and benchmark datasets. Routing analyses characterize information released by the stated protocols; they are not tests on private user traffic. The guarantees assume a trusted preprocessing dealer and at most one semi-honest corrupted server. They do not establish protection against collusion, malicious execution, or arbitrary implementation side channels.

REPRODUCIBILITY STATEMENT

Appendix C documents the measurement methodology, numerical checks, policy evaluations, and additional results. Appendix A specifies the algebra and security assumptions. Plaintext routing-policy evaluations are distinguished from two-party cryptographic executions.

REFERENCES

Amit Agarwal, Elette Boyle, Nishanth Chandran, Niv Gilboa, Divya Gupta, Yuval Ishai, Mahimna Kelkar, and Yiping Ma. Secure sorting and selection via function secret sharing. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 3023–3037. ACM, 2024. doi: 10.1145/3658644.3690359.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2357–2367, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1245. URL https://aclanthology.org/N19-1245/.

Jacob Austin, Augustus Odena, Maxwell Nye, et al. Program synthesis with large language models. arXiv preprint arXiv:2108.07732, 2021.

Donald Beaver. Efficient multiparty protocols using circuit randomization. In *Advances in Cryptology – CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pp. 420–432. Springer, 1992. doi: 10.1007/3-540-46766-1_34.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In *Advances in Cryptology – EUROCRYPT 2015*, volume 9057 of *Lecture Notes in Computer Science*, pp. 337–367. Springer, 2015. doi: 10.1007/978-3-662-46803-6_12.

Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 1292–1303. ACM, 2016. doi: 10.1145/2976749.2978429.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1300.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. arXiv preprint arXiv:1803.05457, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168, 2021.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1280–1297, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.70.

Ruyi Ding, Tianhong Xu, Xinyi Shen, Aidong Adam Ding, and Yunsi Fei. MoEcho: Exploiting side-channel attacks to compromise user privacy in mixture-of-experts LLMs. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 2159–2173. ACM, 2025. doi: 10.1145/3719027.3765174.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness. Zenodo, version 0.4.3, 2024.

Kanav Gupta, Deepak Kumaraswamy, Nishanth Chandran, and Divya Gupta. LLAMA: A Low Latency Math Library for Secure Inference. *Proceedings on Privacy Enhancing Technologies*, 2022(4):274–294, October 2022. ISSN 2299-0984. doi: 10.56553/popets-2022-0109.

Kanav Gupta, Neha Jawalkar, Ananta Mukherjee, Nishanth Chandran, Divya Gupta, Ashish Panwar, and Rahul Sharma. SIGMA: Secure GPT inference with function secret sharing. *Proceedings on Privacy Enhancing Technologies*, 2024(4):61–79, 2024. doi: 10.56553/popets-2024-0107.

Meng Hao, Hongwei Li, Hanxiao Chen, Pengzhi Xing, Guowen Xu, and Tianwei Zhang. Iron: Private inference on transformers. In *Advances in Neural Information Processing Systems*, volume 35, pp. 15718–15731. Curran Associates, Inc., 2022. doi: 10.52202/068431-1143.

Jamie Hayes, Ilia Shumailov, and Itay Yona. Buffer overflow in mixture of experts. arXiv preprint arXiv:2402.05526, 2024.

Chi-Yao Huang, Khoa Vo, Aayush Atul Verma, Duo Lu, and Yezhou Yang. Domain expansion: A latent space construction framework for multi-task learning. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=b95RCZQB29.

Neha Jawalkar, Kanav Gupta, Arkaprava Basu, Nishanth Chandran, Divya Gupta, and Rahul Sharma. Orca: FSS-based secure training and inference with GPUs. In *2024 IEEE Symposium on Security and Privacy (SP)*, pp. 597–616. IEEE, 2024. doi: 10.1109/SP54263.2024.00063.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts. arXiv preprint arXiv:2401.04088, 2024.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2567–2577, 2019.

Andes Y. L. Kei and Sherman S. M. Chow. SHAFT: Secure, handy, accurate and fast transformer inference. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2025. doi: 10.14722/ndss.2025.242287.

Zhengyi Li, Kang Yang, Jin Tan, Wen-jie Lu, Haoqi Wu, Xiao Wang, Yu Yu, Derun Zhao, Yancheng Zheng, Minyi Guo, and Jingwen Leng. Nimbus: Secure and efficient two-party inference for transformers. In *Advances in Neural Information Processing Systems*, volume 37, pp. 21572–21600. Curran Associates, Inc., 2024. doi: 10.52202/079017-0680.

Wen-jie Lu, Zhicong Huang, Zhen Gu, Jingyu Li, Jian Liu, Cheng Hong, Kui Ren, Tao Wei, and WenGuang Chen. BumbleBee: Secure two-party inference framework for large transformers. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2025. doi: 10.14722/ndss.2025.230057.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1260.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah Smith, Pang Wei Koh, Amanpreet Singh, and Hanna Hajishirzi. OLMoE: Open mixture-of-experts language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Nam V. Nguyen, Thong T. Doan, Luong Tran, Van Nguyen, and Quang Pham. LibMoE: A library for comprehensive research on mixture of experts in large language models. *Transactions on Machine Learning Research*, 2026. URL `https://openreview.net/forum?id=PB2ju8tq0n`.

Amir Nuriyev and Gabriel Kulp. Expert selections in MoE models reveal (almost) as much as text. arXiv preprint arXiv:2602.04105, 2026.

Qi Pang, Jinhao Zhu, Helen Möllering, Wenting Zheng, and Thomas Schneider. BOLT: Privacy-preserving, accurate and efficient inference for transformers. In *2024 IEEE Symposium on Security and Privacy (SP)*, pp. 4753–4771. IEEE, 2024. doi: 10.1109/SP54263.2024.00130.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1454.

Bowen Shen, Yuyue Chen, Peng Yang, Bin Zhang, Xi Zhang, and Zoe L. Jiang. SecMoE: Communication-efficient secure MoE inference via select-then-compute. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(30):25286–25294, 2026. doi: 10.1609/aaai.v40i30.39721.

Don Tuggener, Pius von Däniken, Thomas Peetz, and Mark Cieliebak. LEDGAR: A large-scale multi-label corpus for text classification of legal provisions in contracts. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pp. 1235–1241, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Itay Yona, Ilia Shumailov, Jamie Hayes, and Nicholas Carlini. Stealing user prompts from mixture of experts. arXiv preprint arXiv:2410.22884, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1472.

Yancheng Zhang, Jiaqi Xue, Mengxin Zheng, Mimi Xie, Mingzhe Zhang, Lei Jiang, and Qian Lou. CipherPrune: Efficient and scalable private transformer inference. In *International Conference on Learning Representations (ICLR)*, 2025.

Yifan Zhou, Tianshi Xu, Jue Hong, Ye Wu, and Meng Li. CryptoMoE: Privacy-preserving and scalable mixture of experts inference via balanced expert routing. In *Advances in Neural Information Processing Systems*, volume 38, pp. 57480–57503. Curran Associates, Inc., 2025.

Yufan Zhu, Chao Jin, Khin Mi Mi Aung, and Xiaokui Xiao. EncFormer: Secure and efficient transformer inference over encrypted data. arXiv preprint arXiv:2604.09975, 2026.

A Protocol Interfaces, Assumptions, and Proofs

A.1 Reference Computation and Arithmetic

The functionality evaluates a specified fixed-point model circuit, including its approximation and truncation rules, gate coefficients, shared experts, and deterministic current-slot tie order. Its public parameters include the architecture, tensor dimensions, token count, layer ordering, output interface, and the actual public checkpoint for VeilMoE-O. The public schedule specifies operation and message shapes; VeilMoE-C's selected bank addresses instead belong to its declared leakage. Private-weight execution also fixes the owner's layer permutations. The output belongs only to the user; the server adversary's allowed routing release is Eq. (5). A layer boundary is *tie-free* when the k-th and $(k+1)$-st largest router scores differ in the arithmetic used for selection. Ties entirely inside or outside the selected set do not affect set equivalence.

An opening means that the two servers exchange additive or Boolean shares and reconstruct by modular addition or XOR. A public masked value $\hat{v} = v + r_v$ is distinct from an additive share $\langle v \rangle_b$: both servers know the former, whereas neither alone knows v or the complete mask. A mask-dependent FSS key is a program-evaluation key, not an additive share that can be reconstructed. Arithmetic correlation shares are separate objects and are supplied where an operation explicitly requires them.

Value arithmetic uses $\mathbb{Z}_{2^{50}}$. The oblivious extractor uses $\mathbb{Z}_{2^{64}}$: comparison bits are converted directly to 64-bit arithmetic shares, accumulated, and independently masked before the rank opening. This does not sign-extend secret 50-bit scores. A selector is 0 or 1, so locally reducing both of its 64-bit additive shares modulo 2^{50} preserves their sum exactly. Comparisons assume the backend's valid signed range; in particular, sign-based comparison of a difference requires that the unmasked difference not overflow its signed range. Nonlinear approximation and truncation remain those of the specified reference circuit (Gupta et al., 2022; 2024).

Native equivalence and its boundary. With $(P_\pi)_{e,s} = \mathbf{1}\{s = \pi(e)\}$, shuffled slot s contains both score $g_{\pi^{-1}(s)}$ and expert $W_{\pi^{-1}(s)}$. At a tie-free boundary, the native and shuffled selected sets satisfy $\text{Top-}k(gP_\pi) = \pi(\text{Top-}k(g))$. Reindexing the selected weighted sum therefore preserves the layer computation, subject to the same arithmetic and gate convention. Under a boundary tie, this native-set equality need not hold. For $N = 2, k = 1$, equal scores and an expert swap, the smaller-slot rule selects the opposite native expert from a smaller-native-index rule. The implementation still retains exactly one contribution, but the two outputs may differ. Thus a valid drop-free Top-k set and native tie-breaking equivalence are separate claims.

A.2 Masked Linear and Gather Interfaces

For $v \in \mathbb{Z}_{2^\ell}^{1\times d_{\text{in}}}$ and $L \in \mathbb{Z}_{2^\ell}^{d_{\text{in}}\times d_{\text{out}}}$, let $a_{\text{raw}} = vL$ before fixed-point rescaling, with $\hat{v} = v + r_v$ and $\hat{L} = L + r_L$. The masked-linear identity is

$$a_{\text{raw}} = \hat{v}\hat{L} - \hat{v}r_L - r_v\hat{L} + r_v r_L. \tag{6}$$

One concrete additive realization gives server b private contributions $\langle r_v\rangle_b$, $\langle r_L\rangle_b$, and $\langle r_v r_L + r_{a_{\text{raw}}}\rangle_b$. It locally computes

$$u_b = \mathbf{1}\{b=0\}\hat{v}\hat{L} - \hat{v}\langle r_L\rangle_b - \langle r_v\rangle_b\hat{L} + \langle r_v r_L + r_{a_{\text{raw}}}\rangle_b. \tag{7}$$

Opening $u_0 + u_1$ gives $vL + r_{a_{\text{raw}}}$, not vL. The next truncation gate consumes this masked product and produces the correctly rescaled value under its own output mask. The implementation can serialize these contributions inside operation-specific preprocessing; serialization does not turn an FSS key into an arithmetic share. No server receives a complete operand mask. With a public L, the same interface needs the preprocessed activation-mask product but no private-weight correction.

For VeilMoE-O, linearity gives $\sum_b \langle \widetilde{W}_j^p\rangle_b = \sum_e q_{j,e} W_e^p$ without communication. The activation-mask correction is also exact:

$$\tau_j^p + \sum_e \delta_{j,e} T_e^p = \sum_e r_{q,j,e} T_e^p + \sum_e (q_{j,e} - r_{q,j,e}) T_e^p = r_v \widetilde{W}_j^p. \tag{8}$$

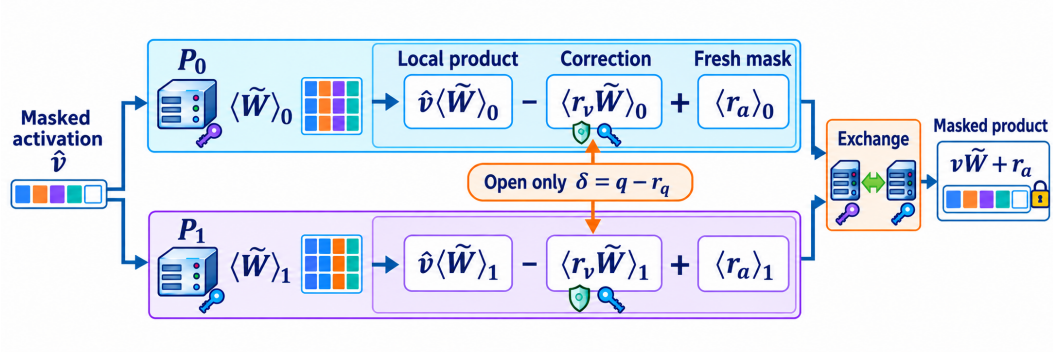

Figure 5: One gathered linear stage before truncation. Each server subtracts its share of the activation-mask product; only a masked selector and the freshly masked product are opened. Projection and rank indices are omitted.

Each stage uses the mask of its actual input activation. Gate and up projections can share the original input activation; each selected down projection has its own preceding SwiGLU activation and corresponding correlations. Output masks, which are added before reconstruction, are independent of the unmasked projection.

The measured one-pass VeilMoE-CG realization prepares a persistent share bank $\langle W'^p\rangle$, a fresh selector mask A, and $\langle AW'^p\rangle$. Since $\Delta = Q - A$ is public,

$$\sum_b \left(\Delta\langle W'^p\rangle_b + \langle AW'^p\rangle_b\right) = (Q-A)W'^p + AW'^p = QW'^p. \tag{9}$$

Neither matrix shares nor the unmasked selector are reconstructed. Reusing one selector mask for correlated operations on the *same* selector is possible with jointly generated material; reusing it for distinct selectors would expose their difference and is disallowed. This one-pass share-bank construction is the measured CG path, rather than an additional online masked-bank gather followed by residual-weight-mask removal.

For CG's activation correction, the trusted preprocessor uses the actual private, permuted matrices to form $T_e^p = r_v W_e'^p$ and $\tau_j^p = \sum_e r_{q,j,e} T_e^p$. It supplies shares of these quantities, not the plaintext products, to the servers. Thus the correction identity above applies with $\widetilde{W}_j^p = \sum_e q_{j,e} W_e'^p$; it does not require either server to possess a public copy of the private bank. The mask r_v is that of this projection's input, including the distinct selected SwiGLU activations used by down projections.

Lemma (single-opening gather privacy). Over a finite ring, let Q, B be shared matrices of compatible dimensions. Suppose preprocessing samples A uniformly independently of these inputs and independently splits A and $C = AB$ into additive shares. Opening $\Delta = Q - A$ and returning $Z_b = \Delta B_b + C_b$ realizes an additive sharing of QB. For one semi-honest server, the joint view of this subprotocol is perfectly simulatable from its input shares Q_b, B_b and output share Z_b. The statement conditions only on prior state independent of the fresh mask and splitting randomness.

Proof. Correctness follows from $\Delta B + C = QB$. For privacy, the simulator samples Δ and A_b uniformly, sets $C_b = Z_b - \Delta B_b$, and supplies the other server's opening message as $\Delta - (Q_b - A_b)$. In the real distribution, Δ is uniform, A_b is independently uniform, and the independent split of C makes C_b uniform. Consequently $Z_b = \Delta B_b + C_b$ is uniform conditional on Δ, A_b, Q_b, B_b. Conditioning on the supplied Z_b therefore gives exactly the simulator's joint distribution, including both preprocessing shares and the opening message, not merely a uniform marginal for Δ. □

Taking $B = W'^p$ gives the CG matrix gather. Taking $Q = q_j$ and B to stack the T_e^p gives the activation-mask correction. Several banks can be concatenated into B to use the same A for the same Q, provided their product tensors receive independent splits. This lemma is a stand-alone correlation-interface argument. It does not assume that separately generated FSS keys and other gates sharing its wires are independent.

A.3 Conditional Composition Argument

For composition, let $\mathcal{F}_{\text{corr}}$ denote the trusted correlation interface. Given the operation schedule, wire-mask identifiers, and the bank visibility contract, it jointly generates FSS keys, arithmetic mask contributions, Beaver material, and the gather correlations above. References to the same wire use the same mask; independent operations use fresh masks and independent splits unless a same-selector reuse is explicitly specified. For CG it additionally receives the private permuted bank; for O it uses the public bank. It delivers only the designated party components and the owner/user boundary masks. The composition claim assumes that each invoked primitive is secure with this joint interface, not merely secure under an unrelated stand-alone preprocessing distribution.

Proposition 1 (Primitive composition)**.** *In the $\mathcal{F}_{\text{corr}}$ hybrid and under the primitive assumptions of Theorem 1, the composed protocol realizes the specified fixed-point functionality with the full membership stream $\mathcal{L}_C$ for VeilMoE-C and no additional routing release for VeilMoE-O and VeilMoE-CG.*

Proof. Work in the hybrid model in which masked linear evaluation, FSS comparison, B2A, truncation, nonlinear gates and Beaver multiplication are secure private interfaces with their specified output representations and compatible joint preprocessing. Local rank accumulation, cumulative-threshold differencing, and public linear combinations compute exactly their stated shared outputs. The gather identities above establish the additional selection-dependent corrections. Hence the output is that of the reference circuit.

The only unmasked routing release is VeilMoE-C's membership. A simulator given the public schedule and $\mathcal{L}_C$ supplies that release and the corresponding bank addresses. For the gather variants it supplies the fixed full-bank scan. The lemma simulates each new gather opening together with its designated correlation shares, conditional on the connected input and output shares. The assumed primitive simulators handle the remaining openings, including masked comparison bits and Beaver operand differences. Their compatibility with $\mathcal{F}_{\text{corr}}$ preserves shared wires and preprocessing dependencies between operations. Composition in this hybrid then yields computational indistinguishability of the complete server view. This argument does not independently replace correlated keys, shares and openings with unrelated uniform samples. □

This is a conditional hybrid-model guarantee, not a proof that arbitrary correlation generation or serialization realizes $\mathcal{F}_{\text{corr}}$. The premise includes the underlying FSS and masked-arithmetic security assumptions (Boyle et al., 2015; 2016; Beaver, 1992; Gupta et al., 2022; 2024). It is not a theorem of security against a corrupted dealer, malicious deviation, colluding servers, or arbitrary implementation side channels. The benchmark harness's local dealer emulation measures operation material and generation time; it is not a role-isolated production deployment or an implementation-security audit.

A consequence of the full stream. Within one layer and permutation epoch, let z_t indicate the actually executed native membership and $z'_t = P_\pi^\top z_t$. Then

$$G' = \sum_t z'_t z'^\top_t = P_\pi^\top \left(\sum_t z_t z_t^\top \right) P_\pi = P_\pi^\top G P_\pi. \tag{10}$$

Every permutation-invariant statistic of G is therefore preserved. The converse is not an information bound: the stream exposes ordering and recurrence beyond G'. For example, memberships $(0, 0, 1)$ and $(0, 1, 0)$ at $k = 1$ give the same G' but different adjacency patterns. Simulation allows any inference from the authorized stream, including linkage or an inference strengthened by auxiliary information. It does not imply a universally random native-identity recovery rate.

A.4 Dealer Lifecycle and Deployment Boundary

Setup and reuse. The model owner provisions the model representation required by the chosen variant. The trusted preprocessing service generates masks and correlated material and then leaves the online protocol. Complete boundary masks $r_{x_{\mathrm{emb}}}$ and $r_{y_{\mathrm{out}}}$ have the shapes of the submitted post-embedding activation and returned model output. They go to the user, who publishes $\hat{x}_{\mathrm{emb}} = x_{\mathrm{emb}} + r_{x_{\mathrm{emb}}}$ and finally subtracts $r_{y_{\mathrm{out}}}$ from $\hat{y}_{\mathrm{out}}$. The specified interface starts after embedding; it does not by itself implement a private vocabulary lookup or tokenize private text remotely. Private-weight masks go to the model owner for masked-bank publication. FSS keys and operation-specific arithmetic shares go to the appropriate servers, never complete masks.

VeilMoE-C can generate its random mask correlations without plaintext private weights. VeilMoE-O's weights are public. In the measured CG implementation, the preprocessing API additionally consumes plaintext private weights to generate AW and activation-mask products. This is an explicit additional trust exposure to the already trusted service, not exposure to an online server. Model-blind CG correlation generation would require a separately implemented and measured setup protocol.

Persistent bank state is tied to a model version and permutation epoch. Per-token wire masks, selector masks, FSS keys and multiplication material are consumed once. A model change requires regenerating the corresponding bank and dependent material. An unavailable dealer need not interrupt already provisioned inference, but new requests cannot continue indefinitely after the fresh material is exhausted. The offline measurements quantify material and generation cost, not an implemented resilient key-management service.

Recovery and compromise. A safe deployment must assign consumable preprocessing identifiers and record consumption durably. After an uncertain interruption, it must burn uncertain material instead of replaying masks or triples on another input. These are deployment requirements, not tested recovery guarantees of the artifact. A compromised dealer holding masks and the relevant masked transcripts can recover protected values. All three variants retain this trust assumption; CG removes membership leakage, not dealer compromise. We do not claim forward secrecy or verified erasure.

Refreshing masks alone leaves VeilMoE-C's membership stream unchanged. Refreshing a permutation changes direct slot identities between epochs but does not prove statistical unlinkability of the epochs. Within an epoch, anonymous recurrence persists. Deployments that cannot tolerate this information use the fixed-access variants. The modeled guarantee does not cover packet arrival patterns, GPU power observations, or other hardware side channels absent from the specified transcript and schedule.

B Routing and Numerical Fidelity Audit

Scope and sampling. We audited the cached OLMoE and DeepSeekMoE checkpoints on the first four eligible documents returned by the fixed loader for each of GSM8K, MBPP, WikiText-2, LEDGAR, and PubMedQA (Cobbe et al., 2021; Austin et al., 2021; Merity et al., 2017; Tuggener et al., 2020; Jin et al., 2019). Each document was truncated to 128 tokens, yielding 1,565 and 1,592 tokens, respectively. Across 16 and 27 routed layers, these constitute 25,040 and 42,984 routing decisions. The native models used bfloat16 weights and teacher-forced inputs; hooks captured the actual gate inputs and selected indices. This is a router-only sensitivity audit, not encrypted execution, task accuracy, or autoregressive error propagation.

Separating numerical references. For fractional precision $f \in \{12, 14\}$, we separately rounded each gate weight and input after multiplication by 2^f, accumulated their integer products, and applied a signed right shift by f. These fractional precisions are distinct from the protocol's ring width

$\ell = 50$; increasing f does not mean changing ℓ. Observed accumulator magnitudes were below 2^{32}. We also computed an unquantized FP64 gate projection using the *same* captured inputs and stored weights, with smaller native index breaking ties. This widened reference isolates quantization from the native gate's arithmetic and tie conventions; it is not a separate FP64-model execution. Native checkpoint selections were recorded separately rather than treated as identical to that reference.

Table 5: Routing counts over layer–token decisions. Q is index-stable quantized selection and R its widened reference; slot changes use one fixed, seeded permutation per layer.

Model	f	Decisions	Boundary ties	Slot-rule changes	$Q \neq R$
OLMoE	12	25,040	28	13	249
	14	25,040	5	4	43
DeepSeekMoE	12	42,984	57	30	251
	14	42,984	15	9	44

Quantization-specific set changes decrease with precision (Table 5). Native checkpoint selections differ from R on 173 OLMoE and 602 DeepSeekMoE decisions; they differ from Q on 297/173 and 625/604 decisions at $f = 12/14$. These are reference-arithmetic differences, not measured cryptographic failures. Decisions share documents and layers, so we do not treat them as independent statistical trials.

Boundary ties. An exhaustive CPU test enumerated all 3^4 score vectors over $\{0, 1, 2\}$ and all 24 permutations for $N = 4, k = 2$. Among 1,944 cases, 936 have a boundary tie; 588 tied cases change the selected set, whereas no tie-free case does. For scores $(0, 0, 0, 0)$, native-index ordering selects $\{0, 1\}$, but the slot-to-native permutation $(0, 2, 1, 3)$ selects $\{0, 2\}$. Equal router scores do not imply equal expert outputs: changing the tied expert can change the MoE output despite retaining exactly k experts and dropping no assignment. The measured slot rule therefore does not establish unconditional native tie equivalence.

Existing live-protocol checks. The real-weight OLMoE check independently supplies captured input to each layer–token block, then substitutes its outputs into a plaintext backbone. Its 16-token results are reported in Table 6; different router/output precisions prevent interpreting them as bitwise equivalence. They do not test accumulated encrypted states or long generation.

Table 6: Real-weight OLMoE validation on 16 tokens. Final logits use a plaintext-backbone splice; block and logit errors are absolute differences.

Quantity	Statistic	Value
Selected expert set	Agreement	100%
MoE-block error	Mean	0.0007485
	Maximum	0.84625
Final-logit difference	Mean	0.17415
	Maximum	1.32062
Next-token argmax	Agreement	100%
Next-token Top-5 set	Agreement	50%

Possible tie-rule correction. Preserving a specified native-index rule requires comparing scores together with *secret* original-index priorities; publishing those priorities would disclose the permutation. Lexicographic secure comparison is one possible correction. Packing both fields into an integer additionally requires a range and bitwidth analysis. Neither correction is implemented or benchmarked here, and an index-stable rule must not be conflated with an unspecified framework's tie behavior.

Provenance. The artifact's routing-audit directory retains per-document hashes, checkpoint revisions, layer-level counts, both audit passes, and source hashes; its aggregation script recreates the

summary. The fixed sampling rule and seed 20260925 precede the measurements; all sampled documents and outcomes are retained.

C Experimental Scope and Extended Results

C.1 What Each Measurement Establishes

Recorded timing hosts each use one A100-SXM4-80GB GPU, dual AMD EPYC 7532 processors and 503 GB RAM. The CUDA C++ backend extends the implementations of Gupta et al. (2022; 2024). Its recorded environment is Ubuntu 22.04, NVIDIA driver 570.148.08, CUDA 11.8, GCC 9.5, CMake 3.22 and Eigen 3.4. The policy evaluations use Python 3.10, PyTorch 2.4.1 with CUDA 12.4, Transformers 4.46.3 and LM Evaluation Harness 0.4.9.1 (Gao et al., 2024). The secure cost harness uses 50-bit model values, with 14 fractional bits for FFNs and 12 for router, softmax and output values. The 64-bit ranking interface is specified separately in Appendix A.1.

The timing harness executes an architecture-sized secure Transformer stack with zero attention and expert banks and a randomized router. These are measured cryptographic executions, not sums of extrapolated per-layer times, but they are not encrypted language-model generations with checkpoint weights. Numerical tests and plaintext task evaluation use separate, nonzero inputs and weights. The artifact's measurement-scope manifest records this distinction for every result family. Dealer emulation occurs locally before the timed online phase; it does not establish physical isolation between the dealer and a computing party.

Appendix B separately reports the teacher-forced checkpoint routing audit and existing live block checks, including numerical maxima. The small nonzero CG test covers 32 output elements, not 32 independent model executions. None of these checks demonstrates long-generation equivalence or a KV-cache serving implementation.

C.2 Routing-Policy Accuracy and Paired Uncertainty

The policy study executes quantized plaintext MoE blocks inside the native model backbone. General tasks are zero-shot ARC-Challenge, ARC-Easy, BoolQ, HellaSwag, OpenBookQA, PIQA, SocialIQA and WinoGrande (Clark et al., 2018; 2019; Zellers et al., 2019; Mihaylov et al., 2018; Bisk et al., 2020; Sap et al., 2019; Sakaguchi et al., 2020). The dedicated mathematical-reasoning task is MathQA (Amini et al., 2019). All policy cells use batch size 16 and a uniform $f = 14$ MoE oracle, distinct from the mixed-precision secure cost harness. Its FP64 nonlinear functions are requantized; they are not bit-exact substitutes for the secure LUT gates. For T processed token positions, a capacity policy keeps at most $\lceil cTk/N \rceil$ assignments per expert, prioritizing larger gate coefficients when capacity is exceeded. The Top-1 policy selects a single expert with unit weight. These reproduce routing restrictions for comparison; they do not execute SecMoE's or CryptoMoE's cryptographic protocols. Results therefore measure sensitivity to these policy changes, not an accuracy ranking of independently trained systems.

General-task accuracy is the unweighted mean of eight task accuracies from 22,419 aligned samples. MathQA uses 2,985 samples and PIQA 1,838, with the harness's `acc_norm`: choice accuracy using length-normalized continuation log likelihood. PIQA is also in the general suite and is not an independent third suite or a controlled context-length experiment. Paired percentile bootstrap uses 10,000 resamples, seed base 20260725, and identical example indices for the exact and altered policies. General-suite resampling occurs within each task before macro averaging.

C.3 Network and Repetition Accounting

LAN uses 1 Gbps and native sub-millisecond RTT; WAN uses 400 Mbps. The 10, 20, 80 and 160 ms profiles add half the target RTT at each host. Network-shaped executions are serialized because traffic-control state is host-wide. Communication is sent-plus-received traffic measured at one server, divided by 2^{20} for MiB; it is not the sum of both servers' counters. Table 8 reports arithmetic means and sample standard deviations across all accepted observations. The available verified archive contains no comparable 4 ms full-stack cell; we do not interpolate one from neighboring RTT settings.

Table 7: Capacity-policy drop rates and paired 95% confidence intervals for accuracy loss from the native-k reference, in percentage points (pp).

Model	Task	Capacity $c = 1$		Capacity $c = 2$	
		Drop (%)	Loss CI (pp)	Drop (%)	Loss CI (pp)
OLMoE	General	27.32	$[2.33, 3.59]$	7.68	$[0.06, 0.84]$
	MathQA	32.24	$[2.95, 5.93]$	12.78	$[0.74, 3.22]$
	PIQA	31.80	$[5.22, 8.27]$	11.18	$[0.82, 2.99]$
DeepSeekMoE	General	21.77	$[0.97, 2.02]$	4.51	$[-0.31, 0.36]$
	MathQA	31.10	$[2.41, 5.33]$	11.35	$[0.17, 2.11]$
	PIQA	28.52	$[0.49, 3.21]$	8.41	$[-0.76, 1.03]$

Table 8: Four-token architecture-sized latency in seconds, with separate mean and sample-SD rows. Each cell has three runs except DeepSeekMoE Dense-SIGMA at 10 ms, which has five.

Model	Method	Statistic	LAN	10 ms	20 ms	80 ms	160 ms
OLMoE	Dense-SIGMA	Mean	16.17	38.21	49.14	126.84	241.15
		SD	2.17	3.57	0.45	1.96	15.52
	VeilMoE-C	Mean	6.01	18.37	39.15	114.17	214.69
		SD	1.90	0.03	9.76	9.44	8.99
	VeilMoE-O	Mean	8.29	21.60	44.19	122.43	225.05
		SD	0.67	0.09	8.86	10.72	9.28
DeepSeekMoE	Dense-SIGMA	Mean	36.98	81.10	112.89	279.59	503.72
		SD	11.31	12.14	19.85	23.04	24.90
	VeilMoE-C	Mean	12.57	38.38	80.73	238.11	447.83
		SD	5.33	0.04	25.44	20.97	13.51
	VeilMoE-O	Mean	20.17	47.39	86.88	250.18	469.89
		SD	7.12	0.08	23.70	16.65	21.58

The separate DeepSeekMoE VeilMoE-CG four-token LAN measurement has mean 37.47 s and sample SD 0.34 s over three runs.

Displayed token-scaling cells use three runs, except VeilMoE-C at six tokens, VeilMoE-CG at eight tokens, and the isolated Dense-SIGMA 12/14/16-token cells, which use five. The original unbound Dense campaign remains a diagnostic dataset; the replacement campaign binds both parties to the GPU-local NUMA node and retains all five accepted observations per cell. Contention-invalidated attempts are retained separately, not averaged. The scaling figure reuses the primary table's four-token anchors for Dense-SIGMA, VeilMoE-C and VeilMoE-O; the larger-batch values come from the scaling campaigns. The independent original four-token scaling records are retained rather than pooled or deleted. No smoothing enforces a monotonic latency curve.

C.4 Preprocessing Resources

Table 9 separates model-version bank state from fresh four-token material. Storage is per party in GiB (2^{30} bytes), including serialized FSS keys and host-side correlations. Generation time first averages the two party-local generation times for one execution, then reports the mean and sample SD over three executions. It is not a central dealer's production throughput or a measured distribution latency. Private-bank material is reusable under the stated version and epoch; fresh batch material is consumable. Public checkpoint storage is excluded from VeilMoE-O's zero private-bank entry.

C.5 Architecture Coverage and Comparator Scope

The one-token routed-layer communication test instantiates four architecture shapes and keeps each model's k fixed. Dense-SIGMA evaluates all N expert FFNs while retaining that k in its secret selection. Table 10 is a protocol-shape comparison, not a controlled sweep of N: expert widths

Table 9: Per-party preprocessing: bank and fresh four-token material in GiB; generation-time mean and sample SD in seconds, with $n = 3$.

Model	Method	Bank	Fresh	Generation (s)	
				Mean	SD
OLMoE	Dense-SIGMA	48.000	24.095	137.86	5.31
	VeilMoE-C	48.000	5.889	103.73	5.00
	VeilMoE-O	0	5.933	15.69	0.79
	VeilMoE-CG	48.000	29.949	157.87	4.10
DeepSeekMoE	Dense-SIGMA	111.401	52.650	307.82	19.80
	VeilMoE-C	111.401	14.554	225.39	12.50
	VeilMoE-O	0	14.631	28.91	2.06
	VeilMoE-CG	111.401	56.398	350.35	7.04

and k also vary. Larger checkpoints are not claimed to have complete encrypted-model timing measurements. The older eight-layer latency grid uses a different dense implementation and is not merged into this strict native-k comparison.

Table 10: One-token routed-layer traffic in MiB. Counts are deterministic and cover routed experts, not the full model.

Architecture	N	k	Traffic (MiB)		
			Dense-SIGMA	VeilMoE-C	VeilMoE-O
Mixtral	8	2	17.779	4.424	4.447
Phi-3.5	16	2	18.696	2.280	2.341
DeepSeekMoE	64	6	22.609	2.422	2.597
OLMoE	64	8	18.904	2.212	2.397

SecMoE and CryptoMoE use client–server HE-based or hybrid protocols; VeilMoE uses two non-colluding online servers and trusted preprocessing. Client participation, routing restrictions, preprocessing and cryptographic backends differ. We have not executed a matched HE implementation, and cross-paper reported timings do not establish a relative speedup. Dense-SIGMA is the controlled runtime comparator. The policy study above addresses a separate question and cannot fill this performance-comparison gap. CG also extends trust exposure by allowing its current trusted preprocessor to consume private weights, as specified in Appendix A.4.

C.6 Routing Views and Auxiliary-Information Tests

Leakage tests use 403 OLMoE and 415 DeepSeekMoE document traces from math, code, general, legal and medical corpora, represented by 256-token windows across all routed layers. Anonymous clustering fits KMeans to normalized co-activation features without training labels. Corpus labels are used only to score ARI after fitting, with 1,000 label permutations. The constant-access control has one distinct feature vector: its ARI of zero is a property of the modeled view, not evidence about hardware side channels.

Identity matching uses the co-activation matrix G_h of Section 4. For reference and observed matrices, it sorts eigenvectors by absolute eigenvalue, takes their element-wise absolute values, and uses Hungarian assignment to maximize row-descriptor inner products. This is our diagnostic adaptation of co-activation and spectral-alignment analyses (Nguyen et al., 2026; Huang et al., 2026), not a claimed optimal attack. Its budget study uses 256–8000 tokens per domain and ten sampled subsets per budget in each of five domains. Unrelated-proxy recovery stays near $1/64$. The self-copy control aligns a graph with its own permuted copy and recovers 97.3%–100% of labels. The exact-checkpoint condition instead builds reference and observed graphs from different traces; its lower recovery in Table 11 is not the self-copy test. Table 11 separates native-label recovery from document linkage and labeled domain prediction. Labeled probes use document-grouped held-out folds. The latter scores use different metrics and must not be compared across rows.

Table 11: Routing-view tests with the stated attacker information. Scores are fractions; metrics differ across rows. The first four rows add auxiliary checkpoints or labels.

Test and metric	Information	OLMoE	DeepSeekMoE
Native recovery	Exact checkpoint	0.0744	0.0788
Domain accuracy	Labeled victim traces	1.0000	0.9928
Unseen-corpus accuracy	Labeled alignment	0.6200	0.4122
Cross-model accuracy	Proxy and labels	0.3251	0.2313
Document-link accuracy	Segmented membership stream	0.3046	0.2611
Document-link reference	Row-specific null	0.0029	0.0030