Preempt

Sanitizing Sensitive Prompts for LLMs

Amrita Roy Chowdhury* David Glukhov* Divyam Anshumaan*

Prasad Chalasani Nicolas Papernot Somesh Jha Mihir Bellare

U. Michigan · U. Toronto & Vector Inst. · UW-Madison · Langroid · UC San Diego

NDSS 2026

The Problem: Privacy at LLM Inference Time

User prompt to LLM API:

*"**Kaiser Soze** is **50** years old and earns **$500,000** per year. What is his ideal retirement plan?"*

PII Exposure

SSN, credit card numbers, names sent to untrusted APIs

In-Context Learning

Training examples in prompts shift privacy risk to inference

Lack of Awareness

Users unwittingly disclose sensitive info to LLMs

Samsung, Amazon, Apple, financial institutions & government agencies have banned the use of proprietary LLMs

Why Existing Solutions Fall Short

HE / MPC	16+ min per inference on BERT	**Impractical**
Redaction	Significantly reduce utility	**No utility**
Substitution (Look-Up Table)	Stateful — growing tables per user	**Not scalable**
LLM-based Obfuscation	No formal privacy guarantee	**No guarantee**
Noising Token Embeddings	Type Mismatch, Curse of dimensionality	**Low utility**

Design Goals & Key Properties

Formal Privacy Guarantees

Provable, mathematical guarantees — not ad-hoc heuristics or trust assumptions.

High Utility & Usability

Sanitized prompts should yield near-identical LLM responses to the originals, while being cheap to compute.

Stateless Design

No stored tables or session state — only a secret key is needed to reverse the sanitization.

Regulatory Compliance

Compatible with GDPR and CCPA. No sensitive data retained after a session.

Key Insight: Prompt-Invariant Tasks

When can we sanitize a prompt without hurting the LLM's response?

When the task is invariant to the exact values of sensitive tokens — only their format or approximate magnitude matters.

Translation	`"Kaiser Soze earns $80K" → Translate to French`	Translation quality is identical whether the name is “Kaiser” or “Marcus”.
RAG / Comparison	`"Adam earns $80K, Bob earns $60K. Who earns more?"`	Answer depends on the relative order, not on exact names or dollar amounts.
Financial Advice	`"Patient age 50, salary $500K. Retirement plan?"`	Advice quality is robust to age ±2 years or salary ±$15K.

This insight reveals two types of sensitivity in tokens:

Format-dependent (names, SSNs — only the format matters) and **Value-dependent** (age, salary — approximate magnitude matters)

Two Categories, Two Methods

Category I — Format-Dependent

The LLM's response depends on the token's format, not its specific value.

Names • SSN • Credit Card • Phone • IP

→ **Format-Preserving Encryption (FPE)**

`Kaiser Soze`	→	`Marcus Chen`
`123-45-6789`	→	`847-29-3156`

✓ **Perfect utility — fully reversible with secret key**

Category II — Value-Dependent

The LLM's response depends on the token's approximate numerical value.

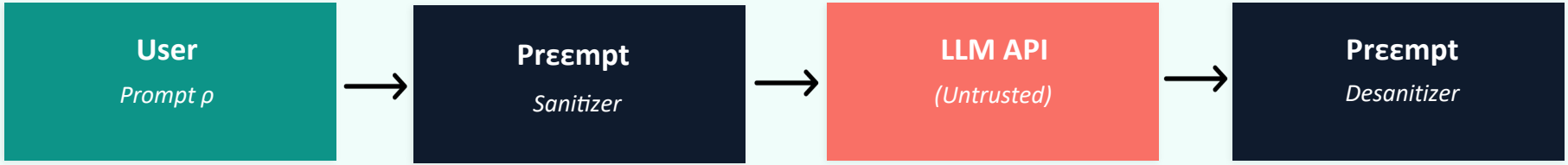
Age • Salary • Bank Balance • Medical Values

→ **Metric Differential Privacy (mDP)**

`Age: 50`	→	`Age: ~48–52`
`Salary: $80K`	→	`Salary: ~$77K–$83K`

✓ **Graceful degradation — utility is preserved**

System Overview



1	**Type Annotation**	NER identifies sensitive tokens and their types (Name, SSN, Age, Salary, ...)
2	**Categorize & Sanitize**	Category I tokens → FPE encryption Category II tokens → m-LDP perturbation
3	**Send to Remote LLM**	Sanitized prompt ρ̂ is sent to the untrusted LLM API
4	**Desanitize Response**	FPE tokens are decrypted; mDP tokens left as-is (stateless!)

Worked Example: Sanitization Pipeline

1. Original Prompt

Kaiser Soze is **50** years old and earns **$500,000** per year.
What is his ideal retirement plan?

2. NER Type Annotation

(**Kaiser Soze**, [Name]) is (**50**, [Age]) years old and earns (**$500K**, [Salary]) ...

Cat I → FPE Cat II → mDP

3. Sanitized Prompt $\hat{\rho}$

Marcus Chen is **48** years old and earns **$485,000** per year.
What is his ideal retirement plan?

FPE: format preserved *mDP: close values*

4. LLM Response

For **Marcus Chen**, at age **48** earning **$485K**/yr, I recommend maximizing 401(k) contributions and ...

5. Desanitized Response

For **Kaiser Soze**, at age **48** earning **$485K**/yr, I recommend maximizing 401(k) contributions and ...

[Name]: perfectly restored via FPE decryption

[Age] & [Salary]: close but not exact (mDP — stateless, no recovery)

Formal Privacy Guarantee

Cryptographic Privacy Game

Adversary picks two prompts (ρ_0, ρ_1) with same leakage profile (using $\mathcal{L}$), differing in a set S of sensitive tokens. It receives one sanitized prompt. The adversary must guess which prompt was chosen for sanitization. Advantage measures distinguishability.

Theorem 2

$$\mathbf{Adv}^{\mathrm{pp}}_{\mathrm{Pr}\epsilon\mathrm{empt},\,\mathcal{L}} \leq e^{l\epsilon} + negl(\kappa)$$

Where $\mathcal{L}$ = leakage function, l = max distance between any τ_{II} tokens in S, ϵ = privacy parameter, κ = FPE security parameter

$e^{l\epsilon}$ from Category II (m-LDP) tokens

$negl(\kappa)$ from Category I (FPE) tokens

Evaluation: Translation & RAG

English → German Translation (BLEU Scores)

Attribute	GPT-4o Plain	GPT-4o Preempt	GPT-4o Papillon
Name	0.287	**0.278**	0.175
Age	0.243	**0.231**	0.135
Money	0.217	**0.200**	0.153

RAG Tasks

100%

accuracy on all tasks

Key Findings

BLEU scores nearly identical
between plain and Preempt-sanitized prompts across all attributes and models

Preempt significantly outperforms Papillon
on translation — especially on [Name] (59% higher BLEU)

RAG task is prompt-invariant
— Comparisons tolerate FPE and m-LDP perturbations

Evaluation: Long-Context & Multi-Turn Q/A

Long-Context Q/A — STS Scores (NarrativeQA)

Metric	Preempt Llama-3	Preempt Gemini-1.5	Preempt GPT-4o	Papillon GPT-4o
vs. Plain	0.839	0.849	**0.934**	0.854
vs. GT	0.514	0.722	0.510	0.458

Multi-Turn Financial Q/A — ConvFinQA

ε	Rel. Error 25th %	Rel. Error Median	Consistency Median
0.5	0.015	0.078	0.135
1.0	0.008	0.041	0.074
2.0	**0.004**	**0.024**	**0.045**

0.934

STS score
GPT-4o + Preempt
vs. plain responses

Utility improves smoothly with larger ε — clear, controllable privacy-utility tradeoff

Why Format Preservation Matters

RAG factual retrieval accuracy with different encryption methods

FPE (Ours)

100%

accuracy

Format preserved

Random Substitution

77.4%

accuracy

Wrong format

AES Encryption

71.0%

accuracy

No format

Format preservation is essential — LLMs rely on format to correctly process structured data

NER: Practical Component Analysis

F1 scores for English (E), German (G), French (F)

	Open-source Models						Closed-source Models					
	Uni-NER-7B-PII			Gemma-2 9B Inst			GPT-4.1			Gemini-2.5		
Attribute	E	G	F	E	G	F	E	G	F	E	G	F
Name	1.00	1.00	1.00	.907	.893	.846	.843	.883	.845	.742	.903	.840
Age	1.00	1.00	1.00	1.00	.951	.990	.970	1.00	.990	.990	.990	.990
Money	.940	.860	.880	.940	.827	.824	.882	.941	.959	.990	1.00	1.00
SSN	.990	1.00	.990	.640	.760	.653	.875	.959	.960	.990	1.00	1.00
CCN	.980	.960	1.00	.952	.962	.873	.971	.971	.980	.980	.990	.970

UniNER matches or outperforms all 6 models across 10 PII types and 3 languages (subset shown). NER is orthogonal to Preempt's privacy guarantee — modeled as ideal functionality. NER failures affect utility, not the soundness of sanitization.

Open Problems & Future Work

Automated Discovery of Token Dependencies

Inferring relationships between sensitive tokens (e.g., "Paris" ↔ "France") without user intervention

Encoding Token Dependencies

Constraining encryption spaces to maintain semantic relationships between related sensitive tokens

Critical Utility Dependencies

Identifying tokens where any sanitization causes unacceptable utility loss (e.g., product codes)

Context-Level Privacy

Protecting sensitive information that emerges from the full prompt context, not just individual tokens

Preempt

Practical prompt sanitization with formal guarantees

- Cryptographically-inspired prompt sanitizer with provable privacy
- Two-pronged approach: FPE for format-dependent, m-LDP for value-dependent tokens
- High utility across translation, RAG, long-context Q/A, and multi-turn tasks
- Stateless design — only a secret key needed, GDPR/CCPA compliant

---

`github.com/danshumaan/preempt`

Thank you! Questions?