

USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

# From Length to Content: Token-Length Side-Channel Attacks on LLM API Merged Outputs

Sijia Li, Tianyu Cui, Miao Chen, Xinjie Lin, and Zheyuan Gu, *Zhongguancun Laboratory;* Xinhao Deng, *Tsinghua University;* Ke Xu and Qi Li, *Tsinghua University and Zhongguancun Laboratory*

https://www.usenix.org/conference/usenixsecurity26/presentation/li-sijia

This paper is included in the Proceedings of the 35th USENIX Security Symposium.

August 12–14, 2026 • Baltimore, MD, USA

ISBN 978-1-939133-58-8

Open access to the Proceedings of the 35th USENIX Security Symposium is sponsored by



جامعة الملك عبدالله للعلوم والتقنية

King Abdullah University of Science and Technology

ARTIFACT EVALUATED
USENIX
AVAILABLE

# From Length to Content: Token-Length Side-Channel Attacks on LLM API Merged Outputs

Sijia Li[†], Tianyu Cui[†], Miao Chen[†], Xinjie Lin[†], Zheyuan Gu[†], Xinhao Deng[§], Ke Xu[§†], Qi Li[§†]

[†]*Zhongguancun Laboratory* [§]*Tsinghua University*

## Abstract

Large Language Models (LLMs) are increasingly accessed via remote APIs over encrypted channels. While previous studies have shown that fine-grained side channels can leak information under token-by-token processing, modern LLM services typically stream outputs in multi-token chunks, seemingly mitigating such token-level leakage. However, in this paper, we demonstrate that token aggregation does not eliminate privacy risks. Instead, it introduces a merged token-length side channel. In this channel, the sizes of encrypted response chunks inadvertently expose character-length patterns of merged token groups. To study the practical implications of this leakage, we propose *PromptEcho*, a passive eavesdropping attack that reconstructs semantically meaningful portions of natural-language responses from observations of merged token lengths. *PromptEcho* frames the reconstruction task as a constrained sequence recovery problem, leveraging semantics-aware reasoning to resolve the ambiguity caused by merged token transmission. Specifically, *PromptEcho* first applies semantics-aware reasoning to decompose merged-token length observations, then uses probabilistic semantic alignment with a fine-tuned language model's linguistic priors to infer the most likely underlying tokens.

We evaluate *PromptEcho* on real-world API interaction traffic from multiple commercial LLM services, correctly inferring the topic for 42.5% of sessions and reconstructing 25.5% of responses with a cosine similarity above 0.9 on OpenAI's GPT-4o and DeepSeek-V3. These findings demonstrate that merged-token transmission remains vulnerable to token-length side-channel attacks, indicating practical privacy risks from encrypted traffic observations under the evaluated API-based LLM settings.

## 1 Introduction

The rapid evolution of Large Language Models (LLMs) has led to their integration into privacy-sensitive domains such as finance [23, 27, 41], medicine [32, 36, 43], law [11, 12, 34], and office assistance [2, 20, 29]. To meet the substantial computational demands, cloud-based APIs have become the primary method for users and developers to access these models. A recent study has examined privacy leakage in LLM interactions via side-channels of response length [38], where one token is processed or transmitted at a time, which can expose sensitive information from each transmitted token.

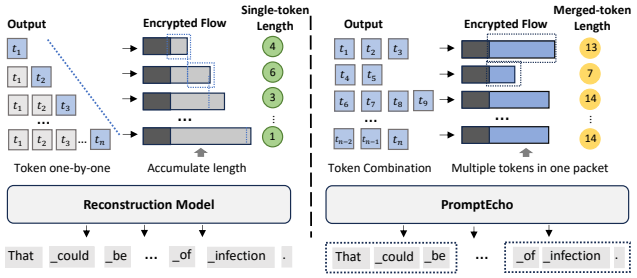



Figure 1: Different structures between single-token streaming and merged-token streaming in LLM API communication.

In response to the threat, LLM APIs have adopted a merged-token streaming mechanism to mitigate privacy leakage from single-token transmission, while balancing transmission efficiency and user experience. As shown in Figure 1, they aggregate multiple tokens into a single network packet, masking individual token boundaries and rendering traditional token-level inference attacks ineffective. However, we find that even with merged transmission, the network packet sizes still reveal the total character length of the underlying merged tokens. This residual packet information creates a merged token-length side channel, allowing adversaries to bypass existing protections and enable semantic inference from encrypted traffic. Crucially, because LLM interactions contain explicit and concentrated sensitive content, such as medical and mental-health conditions or political interests, recovering semantically similar responses, key entities, or topic-level semantics already constitutes meaningful privacy leakage, without perfect verbatim reconstruction.

In this paper, we introduce *PromptEcho*, a token inference attack tailored for merged token transmissions in modern

LLM API interactions. This attack infers semantically plausible text from a vast set of token combinations that satisfy length constraints under uncertain token boundaries. Leveraging the linguistic priors and chain-of-thought reasoning abilities inherent in LLMs, *PromptEcho* facilitates semantics-aware decomposition of merged token lengths and guides contextually consistent response inference. Specifically, we frame the reconstruction process as a reasoning-guided sequence recovery task. We utilize a unified representation that maps merged token lengths into a shared semantic space, and implement a progressive learning framework that evolves from one-to-one single-token alignment to merged token-length inference with reasoning-based decomposition. Ultimately, *PromptEcho* achieves measurable semantic inference performance from merged token-length observations, inferring semantic information from conversational responses across a broad range of evaluated topics and API services.

Through an extensive empirical investigation of real-world API traffic, we observe merged-token length leakage across multiple evaluated LLM platforms. We evaluate *PromptEcho* on OpenAI's GPT-4o and DeepSeek-V3, demonstrating its ability to accurately infer the correct topic in 42.5% of sessions and reconstruct 25.5% of responses with a cosine similarity greater than 0.9 from encrypted API streams. Our findings highlight that the attack generalizes across multiple evaluated domains and maintains effectiveness across unseen-topic settings. Additionally, we assess its robustness in challenging environments, including sessions with preamble interference and truncation noise, further suggesting that merged-token transmission poses a non-negligible semantic privacy risk in encrypted LLM deployments under the evaluated conditions.

Our contributions can be summarized as follows:

- **Merged Token-Length Side Channel.** We identify and analyze a previously unexplored leakage vulnerability in LLM APIs with merged-token streaming. Although token aggregation obscures individual token boundaries, encrypted traffic packet sizes still reveal merged token length patterns, enabling semantic inference from streamed responses.

- **Generative Response Reconstruction.** We propose *PromptEcho*, an innovative generative attack powered by reasoning-guided inference. *PromptEcho* incorporates semantics-aware token decomposition to reconstruct semantically consistent response content from the merged token-length side-channel.

- **Cross-Platform Evaluation.** We conduct empirical validation of the attack across multiple AI services, including OpenAI's ChatGPT and DeepSeek, demonstrating consistent semantic leakage across diverse evaluated interaction scenarios and API services.

Table 1: Characterization of network transmission, token merging patterns, and corresponding attack effectiveness across various LLM API Services.

| API Service | Stream | TLS | Merged | $N_t$ | Vuln | ASR |
|---|---|---|---|---|---|---|
| Gemini [20] | ✓ | ✓ | × | 1 | HIGH | 55% |
| GLM-V4.5 [4] | ✓ | ✓ | × | 1 | HIGH | 53% |
| ChatGPT [29] | ✓ | ✓ | ✓ | 2∼3 | MED | 43% |
| LLaMA [16] | ✓ | ✓ | ✓ | 1∼4 | MED | 40% |
| DeepSeek [2] | ✓ | ✓ | ✓ | 2∼3 | MED | 36% |
| Qwen [5] | ✓ | ✓ | ✓ | 1∼4 | MED | 30% |
| StepFun-3.5 [3] | ✓ | ✓ | ✓ | 1∼5 | MED | 32% |

**Notes:** $N_t$: Number of tokens aggregated per TLS record; **Vuln**: Vulnerability level to traffic analysis; **ASR**: Attack Success Rate of our method on the target service.

## 2 Background

### 2.1 Tokenization and Ambiguity

LLMs operate on tokens, which are discrete text units produced by a tokenizer. A token may correspond to a word, a subword fragment, punctuation, or a unit that includes leading whitespace. Formally, a tokenizer maps an input string $P$ to a token sequence $[p_1, \ldots, p_n]$, where each $p_i \in V$. $V$ is the vocabulary of the tokenizer.

The catch is that different tokenizers handle the same text in varied ways, which means token splits and total counts can differ quite a bit. A lot of this comes down to specifics like apostrophes in contractions (think *I'll* or *it's*), hyphens in compounds (such as *follow-up*), general punctuation, and how leading whitespace gets treated. For clarity, we mark whitespace as "_" here.

**ChatGPT-4 and DeepSeek-V3:**

I'll _check _if _it 's _a _follow -up _infection .

**Mistral-V0.3 and LLaMA-2:**

I ' ll _check _if _it ' s _a _follow - up _infection .

In modern LLM services, tokenizers are typically available to developers and are not treated as proprietary secrets. Major providers release tokenizer implementations or compatible tooling [6, 7, 14, 19, 30]. This allows us to obtain the exact tokenizers used by the target models in our experiments.

Crucially, tokens do not map to words one-to-one. Therefore, when an adversary observes only length-related signals, mapping observed token lengths back to text is inherently ambiguous. This ambiguity is amplified under merged-token transmission, where multiple tokens are aggregated into a single length observation, making reconstruction substantially more challenging.

## 2.2 API-Based LLM Interaction

Unlike locally deployed language models, API-based LLM services place a network boundary between users and inference, which prompts are processed by a cloud backend and responses are returned over an encrypted channel.

The client submits a prompt $P$ over an encrypted session, tokenized into $(p_1, p_2, ..., p_n)$, and used to generate an autoregressive output according to $p(r_i|P, r_{1:i-1})$, which is detokenized into the response text $R$. To reduce user-perceived latency, responses are typically streamed as incremental chunks, where newly generated tokens are detokenized and packaged instantly and delivered over TLS-protected web protocols (often HTTPS/2 with SSE-style streaming).

While $R$ is encrypted, TLS record sizes and ordering for streamed chunks remain observable, enabling packet-length side-channel inference. Table 1 summarizes vendor behaviors and the aggregation details.

# 3 Attack Overview

## 3.1 Threat Model

**Attack Scenario.** We consider a remote API-based LLM service where users exchange prompts and responses over encrypted channels. The victim interacts with the API-based LLM service through a public network, while the adversary is situated within the communication path between the client and the LLM service server, acting as a passive on-path observer for traffic monitoring.

**Adversary's Capabilities.** The adversary monitors encrypted traffic between the client and the LLM service, observing the sizes of TLS records in the encrypted response stream. However, the adversary cannot decrypt payloads, modify or inject traffic, or access internal model states (e.g., next-token probabilities). As a result, the adversary's view is limited to the encrypted response stream and its observable transmission patterns. This passive network observer is standard in prior traffic analysis work [13, 17, 21, 25, 28, 40, 42] and corresponds to practical on-path monitoring (e.g., ISP or shared-network adversaries). Many LLM downstream integrations and developer-facing APIs rely on Server-Sent Events (SSE) or WebSockets for real-time response, which naturally expose observable response token size variations and packet boundaries to the on-path observer. The adversary may further collect auxiliary data from public sources or by querying the target service through standard API access to train or calibrate inference models, without privileged access.

**Attack Objective.** The adversary aims to assess potential semantic privacy risks by inferring sensitive semantic information from observable merged token-length patterns in encrypted traffic. The attack seeks to reconstruct partial response information, such as semantically similar content, key entities, or topic-level semantics under the considered threat model.

## 3.2 Merged Token-length Side-channel

By examining how modern API-based LLM services deliver streamed responses in Section 2.2, we identify a merged token-length side channel. Although response contents are protected by TLS, an on-path observer can still measure the sizes and ordering of TLS records that carry the streamed chunks. These observable sizes reveal structured merged length signals about the generated token groups, which can be exploited to infer semantically consistent responses and private information.

To improve transmission efficiency and responsiveness, providers typically stream outputs as chunks. Our measurements across multiple vendors (Table 1) show that tokens are frequently aggregated before transmission. A single TLS record may carry multiple consecutive tokens (e.g., 2∼4 or even larger groups in some deployments). This aggregation changes the leakage granularity from single-token lengths to *group-wise* lengths, but it does not remove leakage.

Let the response be a token sequence $R = (r_1, \ldots, r_m)$. Under merged transmission, the adversary observes an aggregated length sequence $\mathcal{T}_{merged} = (t_1, \ldots, t_n)$, where each $t_j$ corresponds to the total *character length* of a contiguous token span $(r_{k_j}, \ldots, r_{k_{j+1}-1})$, satisfying

$$t_j = \sum_{i=k_j}^{k_{j+1}-1} |r_i|. \tag{1}$$

The grouping boundaries $\{k_j\}$ are latent and not directly observable, making the mapping from $T$ to underlying tokens inherently ambiguous.

Merging coarsens the granularity of token-level observations but maintains structural information about the generated response. Although the token boundaries are obscured, each $t_j$ still imposes a deterministic length constraint on a contiguous span. Together with regularities in syntax, punctuation, and contextual coherence, these constraints sharply restrict the set of plausible token sequences. Consequently, the aggregated length sequence $T$ can provide semantic signals that support topic inference and partial reconstruction of coherent response spans under encryption.

However, it is non-trivial to turn this leakage into a practical attack. Compared to prior single-token settings, the attacker no longer observes explicit token boundaries and must recover plausible text using only aggregated length constraints. We therefore turn reconstruction into two sub-problems. (i) *Merged Length Decomposition.* For each $t_j$, infer the latent token count and boundaries and decompose it into a sequence of per-token lengths that sum to $t_j$. This step is uncertain because a single $t_j$ can arise from many different length partitions. (ii) *Combined Token Derivation.* Given a hypothesized decomposition of merged token lengths, recover a fluent and context-consistent token sequence whose token lengths match the decomposition. Since many different token combinations can satisfy the same length constraint while implying diver-

gent semantics, this step faces semantic ambiguity. Nevertheless, through progressive learning, our reconstruction model learns the underlying correlations between merged lengths and token combinations, thereby achieving semantic response reconstruction in the evaluated merged-token scenarios.

In this paper, we propose an attack workflow that incrementally prunes the reconstruction space using linguistic constraints and LLM-guided reasoning, enabling merged length decomposition and combined token derivation.

## 4 Design of PromptEcho

*PromptEcho* is a reasoning-guided generative attack designed to recover semantically consistent response content from LLMs using merged token-length side channels observed in encrypted API streaming. Merged observations introduce structural ambiguity as each observed length correspond to multiple latent token combinations, so we formulate reconstruction as a length-constrained sequence recovery problem and address it with progressive semantics-aware learning, transferring from deterministic single-token alignment to CoT-guided adaptation under merged observations.

As shown in Figure 2, this design yields an end-to-end pipeline with three modules, which are described in the following subsections.

### 4.1 Length Extraction & Input Design

This section extracts merged token-length observations from encrypted API streaming and converts them into context-aware, segment-level inputs that make reconstruction computationally manageable.

**Extracting Merged Token Lengths.** The attack begins with passively capturing encrypted client–server traffic. As described in Section 3.1, the adversary can observe and intercept traffic by eavesdropping on a shared network (e.g., the same LAN or Wi-Fi), a compromised network router, or an upstream Internet Service Provider, and identify the target response stream by filtering flows based on the provider's server IP addresses and transport protocol (e.g., TCP/TLS).

From the captured encrypted traffic, the adversary observes the sizes and order of TLS records that carry streamed LLM outputs. Since mainstream services aggregate multiple adjacent tokens per record and typically do not add padding or encoding before encryption, each record length equals a service-specific fixed overhead plus a content-dependent payload length. We therefore subtract the fixed overhead to obtain a merged token-length sequence $\mathcal{T}_{merged} = \{t_1, \ldots, t_n\}$. Implementation details on response boundary identification and overhead estimation are provided in Appendix A.1 and A.2.

**Segmentation for Input Design.** Directly reconstructing a long merged token-length sequence $\mathcal{T}_{merged}$ is challenging due to merged observations introducing local ambiguity, where a single observed length may correspond to multiple potential token combinations. These ambiguities accumulate across long responses, rapidly expanding the candidate reconstruction space and increasing the risk of semantic drift during reconstruction. Segmentation is therefore used to improve reconstruction tractability and preserve local contextual continuity. Although it does not eliminate merged-token uncertainty, it localizes ambiguity within shorter spans and reduces error propagation across distant response regions.

To this end, We segment $\mathcal{T}_{merged}$ into shorter structurally coherent spans $\mathcal{S} = \{\mathcal{T}^{(1)}, \ldots, \mathcal{T}^{(S)}\}$ that preserve coarse-grained contextual continuity. Observable cues in the merged-length signal, including punctuation-related patterns, abrupt length transitions, and service-specific chunking, are incorporated into a constrained optimization framework that integrates these cues with statistical length priors. Let $B = \{b_1, \ldots, b_K\}$ denote the boundary indices with $0 = b_0 < b_1 < \cdots < b_K = N$, and define the cumulative length of each segment as $L^{(k)} = \sum_{j=b_{k-1}+1}^{b_k} t_j$. We seek the boundary set $B$ that balances segment-length plausibility, boundary confidence, and segmentation granularity:

$$\widehat{B} = \arg\max_{B} \left( \sum_{k=1}^{K} \log p(L^{(k)}) + \sum_{i \in B} s_i - \lambda K \right), \quad \text{s.t.} \quad L_{\min} \le L^{(k)} \le L_{\max} \tag{2}$$

where $\log p(L^{(k)})$ is a length prior estimated from training data ensuring segment-length plausibility, $s_i$ is a boundary score capturing local structural changes in the length signal, and $\lambda$ controls segmentation granularity by penalizing over-segmentation.

We solve the objective with dynamic programming over a pruned candidate set to limit combinatorial growth under merged-token uncertainty, and apply lightweight template heuristics for common response templates (e.g., list-style headers). Additional details are in Appendix A.3.

After segmentation, we construct context-aware input instances for reconstruction. The first segment $\mathcal{T}^{(1)}$ is processed independently, while each subsequent segment $\mathcal{T}^{(k)}$ is paired with the immediately preceding segment to form a context-enhanced instance $(\mathcal{T}^{(k-1)}, \mathcal{T}^{(k)})$. Notably, the context provided by the preceding segment consists of the merged token-length sequence rather than the already reconstructed text, which captures local sequential dependencies in the response while limiting long-range ambiguity propagation across reconstructed spans. At this stage, each instance includes the current length segment and its preceding length context, providing structural constraints for inference.

### 4.2 Progressive Semantics-Aware Learning

In this section, we present the core training strategy of *PromptEcho* for reconstructing content from merged token-length side channels. Merged transmission introduces two

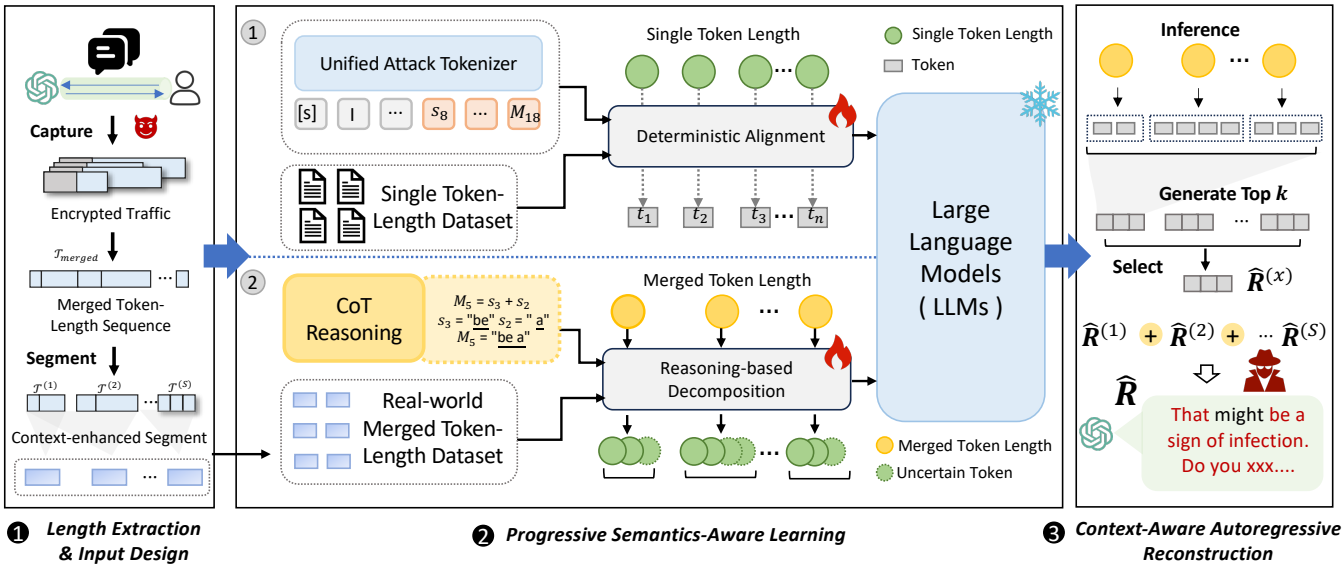



Figure 2: The overall framework of *PromptEcho*.

coupled challenges. First, token boundaries are hidden within each merged length, making it difficult to determine how many tokens compose or where individual token boundaries lie, resulting in token decomposition uncertainty. Second, a single aggregated length can correspond to multiple semantically plausible token sequences, leading to severe semantic ambiguity. These properties render direct reconstruction of content from length information impractical.

To address these challenges, we train a reasoning-guided reconstruction model with a progressive semantics-aware framework that separates semantic learning, which focuses on token-level linguistic patterns, from structural adaptation, which enables the model to adjust to the complexity of merged-token sequences. We first learn a deterministic [*length*, *token*] alignment under single-token supervision, which teaches the model length-conditioned linguistic priors and filters out implausible token hypotheses. We then adapt the model to merged observations, using CoT-guided length decomposition to infer plausible partitions of aggregated lengths while reusing the learned priors to select semantically coherent candidates.

Through this progressive transfer from deterministic to ambiguous supervision, the reconstruction model learns to jointly resolve boundary uncertainty and semantic ambiguity, enabling coherent reconstruction under coarse-grained merged-token observations.

### 4.2.1 Unified Token-Length Representation

We design a unified symbolic representation that maps length observations from both single tokens and merged tokens into the model's input space, while explicitly preserving their respective granularity.

Our progressive learning strategy relies on a token length representation compatible with the model's pretrained linguistic space. However, merged-token signals differ structurally from single-token ones: a single-token length maps deterministically to one token, whereas a merged length aggregates multiple tokens, so conflating the two obscures the source of ambiguity. Directly feeding raw numbers is also unreliable, because pretrained language models often assign natural-language semantics to numerals rather than treating them as token-structure signals.

To address these issues, we design a unified token-length representation that embeds both single-token and merged-token observations into a shared semantic space while explicitly distinguishing their granularity. Specifically, we introduce distinct symbolic forms for single-token length and merged-token length observations, allowing the model to recognize whether an observed length corresponds to a deterministic unit or an aggregated segment.

Concretely, we implement this representation by extending the tokenizer with a set of symbolic length tokens that map numerical length observations into the model's input space. A primitive (single-token) character length $n$ is represented by a dedicated token $s_n$, while an aggregated (merged-token) length of $x$ characters is represented by a separate token $M_x$. For example, a single-token length sequence $\{1,3,5\}$ becomes $\{s_1, s_3, s_5\}$, and a merged-token length sequence $\{8,12,6\}$ is tokenized as $\{M_8, M_{12}, M_6\}$. We introduce primitive length tokens up to $s_{20}$ and merged length tokens up to $M_{60}$, covering 99% of observations. Their embeddings are initialized from a normal distribution rather than tied to numeric tokens, preventing unintended number semantics.

This unified representation preserves the clear token–length correspondence in the single-token setting while explicitly encoding aggregation uncertainty in the merged setting. Dur-

ing both training and inference, the tokenizer serves as the interface between raw length observations and model inputs, ensuring that downstream learning operates on consistent symbolic identifiers rather than ambiguous numeric values.

#### 4.2.2 Deterministic Alignment Fine-Tuning

We first learn length-conditioned linguistic priors under unambiguous supervision, enabling later reconstruction to reject semantically implausible candidates when merged lengths admit many realizations, which mitigate semantic ambiguity in merged-token reconstruction.

We construct single-token training sequences where each observation corresponds to exactly one underlying token. Using the publicly documented tokenizer of the target model, we tokenize each assistant response and compute the character length of every token, producing a length sequence $\{n_1,\ldots,n_k\}$. We then convert it into our symbolic form $\{s_{n_1},\ldots,s_{n_k}\}$ to obtain deterministic, noise-free supervision.

To learn robust priors, we use large-scale public dialogue corpora that provide diverse real-world responses. We fine-tune a pretrained LLM backbone with LoRA, optionally choosing a compatible model/tokenizer family to match the target model's phrasing and tokenization.

Formally, the fine-tuning objective is to reconstruct the original natural-language response from length-based token observations under a given granularity. Given an instruction sequence $I=\{i_1,i_2,\ldots,i_m\}$ and a corresponding length-based sequence $X=\{x_1,x_2,\ldots,x_n\}$, where each $x_i$ denotes a single-token length in this stage, the model predicts the target response $R=\{r_1,r_2,\ldots,r_T\}$ by maximizing the conditional probability

$$\hat{R}=\arg\max_{R} P_\theta(R\mid I,X), \tag{3}$$

where $P_\theta(\cdot)$ represents the conditional probability distribution parameterized by model weights $\theta$. $R$ denotes the ground-truth response and $\hat{R}$ is the predicted sequence generated by the model. Training minimizes the standard autoregressive negative log-likelihood:

$$\mathcal{J}(\theta)=-\frac{1}{T}\sum_{t=1}^{T}\log P_\theta(r_t\mid r_{<t},I,X), \tag{4}$$

where $P_\theta(r_t|r_{<t},I,X)$ represents the probability of generating token $r_t$ conditioned on previous outputs, the instruction prompt, and the length observation sequence. This encourages the model to learn a deterministic alignment between symbolic token-length representations and linguistically plausible token realizations.

This stage learns a stable alignment between length symbols and linguistically plausible token realizations, forming transferable priors that narrow the hypothesis space in the subsequent merged-token setting.

**Merged Token Sequence Training Example**

**Instruction.** Translate the following merged token-length sequence into English:

$$M_{10}\ M_5\ M_8\ M_{11}$$

**Chain-of-Thought.**

- $M_{10}=s_4+s_5\rightarrow$ "That" + "_could"
- $M_5=s_3+s_2\rightarrow$ "_be" + "_a"
- $M_8=s_5+s_3\rightarrow$ "_sign" + "_of"
- $M_{11}=s_{10}+s_1\rightarrow$ "_infection" + "."

**Answer.** That could be a sign of infection.

Figure 3: Training example for the inference model using CoT to decompose merged length sequences.

#### 4.2.3 Reasoning-based Decomposition with CoT

In merged-token settings, token boundaries are unobserved, so the model must infer plausible partitions of each aggregated length instead of using fixed heuristics. We treat boundary inference as a latent reasoning problem and supervise it with CoT during training. Given a merged length token $M_x$, the model is trained to propose a decomposition into primitive length symbols whose sum equals $x$, and then generate the final text conditioned on these internal hypotheses. Figure 3 shows an example of the process.

Building on the model fine-tuned on single-token length sequences, we adapt it to merged-token data collected from real-world API traffic. To preserve the deterministic [*length*, *token*] alignment and linguistic priors learned earlier, we freeze the backbone and update only the embeddings of merged-length symbols $M_x$, together with lightweight LoRA adapters. Each merged length sequence $\mathcal{T}_{\text{merged}}=\{t_1,\ldots,t_n\}$ is mapped into $\{M_{t_1},\ldots,M_{t_n}\}$ using the unified tokenizer described in Section 4.2.1. Each training instance contains a `Chain-of-Thought` segment that explains one plausible decomposition for each $M_x$, followed by an `Answer` segment with the reconstructed text. Formally, the chain represents $M_x$ as a concatenation of primitive symbols:

$$M_x=\text{concat}(s_{n_1},\ldots,s_{n_k}),\ \text{s.t.}\ \sum_{j=1}^{k}n_j=x,\ n_j\in\mathbb{N}_+. \tag{5}$$

We optimize a weighted objective that supervises both the final output and the reasoning chain:

$$\mathcal{J}_{\text{II}}(\theta)=\mathcal{J}_{\text{Answer}}(\theta)+\beta\,\mathcal{J}_{\text{CoT}}(\theta), \tag{6}$$

where $\mathcal{J}_{\text{Answer}}$ and $\mathcal{J}_{\text{CoT}}$ are cross-entropy losses on the `Answer` and `Chain-of-Thought` segments, and $\beta\in(0,1)$ controls the strength of CoT supervision.

This reasoning-guided decomposition transfers the priors learned from abundant single-token supervision to realistic

merged-token observations, enabling robust reconstruction under boundary uncertainty.

### 4.3 Context-Aware Autoregressive Reconstruction

Given a merged token-length trace extracted from encrypted traffic, *PromptEcho* reconstructs the victim response via length-constrained autoregressive decoding, using local context to maintain cross-segment coherence.

We start from the merged length sequence $\mathcal{T}_{merged} = \{t_1, \ldots, t_n\}$ and partition it into $S$ segment $\{\mathcal{T}^{(1)}, \ldots, \mathcal{T}^{(S)}\}$ as in Section 4.1. Each segment is then symbolized as $\{M_{x_1}, \ldots, M_{x_\ell}\}$ using the tokenizer in Section 2.1, producing the model input for reconstruction.

Reconstruction proceeds in an autoregressive manner with the fine-tuned model $\mathcal{M}$. For the initial segment $\mathcal{T}^{(1)}$ which usually conveys the conversation's main intent and topic, generation is conditioned on the symbolic length input together with the prompt-level instruction. For subsequent segments $\mathcal{T}^{(j)}$, generation additionally conditions on the preceding span's length symbols to propagate local context and improve consistency across spans. In this way, reconstruction and context accumulation proceed synchronously rather than in separate stages.

To reduce decoding variance and avoid locally length-compatible but inconsistent hypotheses, we generate $k$ candidates for each segment and select the most likely one under the model:

$$\hat{R}^{(x)} = \underset{i \in \{1,\ldots,k\}}{\arg\max} \log P_\theta \left( R^{(i)} \mid I, \mathcal{T}^{(x-1)}, \mathcal{T}^{(x)} \right), \tag{7}$$

where $I$ is the instruction context and $\mathcal{T}^{(0)} = \varnothing$ for the first segment. We then concatenate the selected outputs to form the final reconstruction:

$$\hat{R} = \hat{R}^{(1)} \,\|\, \hat{R}^{(2)} \,\|\, \cdots \,\|\, \hat{R}^{(S)}. \tag{8}$$

Although CoT supervision is used during training to teach the model how to reason about token decomposition and segmentation, inference does not explicitly generate intermediate reasoning traces. Instead, the learned reasoning behavior is implicitly encoded in the model parameters, enabling direct generation of natural-language outputs that satisfy merged length constraints.

By integrating segmentation-based structural constraints, unified token length representations, and context-aware autoregressive decoding with consistency-based selection, PromptEcho enables reliable and semantically consistent reconstruction of private responses from coarse-grained merged token-length side-channel observations. This unified pipeline narrows the search space, stabilizes outputs, and enforces length-consistent decoding, completing the end-to-end reconstruction attack process under realistic API deployments.

Table 2: Statistics of datasets used in experiments.

| Source | Samples | Avg.Tokens | Type | Usage |
|---|---|---|---|---|
| GPT-4 Turbo | 570,000 | 220 | Single | Alignment |
| GPT-4o | 10,000 | 303 | Merged | Decomposition |
| DeepSeek-V3 | 10,000 | 425 | Merged | Decomposition |

## 5 Evaluation

In this section, we evaluate the practical privacy risks introduced by the merged token-length side channel, demonstrating the effectiveness of *PromptEcho* in reconstructing semantically consistent LLM responses from encrypted traffic. Our evaluation is conducted on LLM API services provided by major vendors, including OpenAI and DeepSeek.

### 5.1 Experimental Setup

**Datasets.** We use two datasets serving distinct purposes in training and evaluating *PromptEcho*. To support deterministic alignment learning, we first construct a large-scale single-token-length dataset derived from real LLM interaction transcripts. We use the UltraChat corpus [15], which consists of real-world human-assistant dialogues that cover 30 meta-topics and 1,100 sub-topics across diverse domains, offering varied conversational structures and realistic linguistic distributions suitable for deterministic token-length alignment learning. From the 570K general-inquiry subset, we extract the first assistant response of each dialogue as a single-turn sample. As these interactions are provided in plaintext form, we tokenize each response using the target model's tokenizer and derive the corresponding single-token-length sequences. This process yields precise one-to-one alignment between tokens and their character lengths, enabling supervised training for deterministic length–token mapping.

To evaluate PromptEcho under realistic deployment conditions, we collect encrypted API traffic from live interactions with commercial deployment, including GPT-4o and DeepSeek-V3, to form merged-token length datasets. We randomly sample 10K prompts from the same general-inquiry subset and issue each query to both services, capturing encrypted HTTPS traffic and reconstructing merged token-length sequences following Section 4.1. Each record corresponds to a real request–response interaction with authentic transmission behavior and encryption overhead. We use 9K samples of each merged-token-length dataset for training the reconstruction model and 1K samples for testing.

The derived single-token-length dataset supports deterministic alignment learning, while the real-world encrypted dataset is used to evaluate merged-token inference performance. We remove any test samples that fully overlap or share the first eight words with training prompts to prevent

data leakage. Dataset statistics are summarized in Table 2. Both training and inference stages use standardized prompt templates, more details are provided in Appendix B.

**Evaluation Metrics.** Since PromptEcho aims to reconstruct semantically consistent LLM outputs and evaluate the resulting privacy risks, we measure performance at three complementary levels. Character- and token-level metrics capture exact textual reconstruction fidelity, while semantic-level metrics assess whether reconstructed outputs expose meaningful user intent or sensitive topics. At the *character level*, we compute Edit Distance (ED) to measure reconstruction accuracy. At the *token level*, we report ROUGE-1 (R1) and ROUGE-L (RL) to measure lexical and structural overlap between reconstructed text and ground truth. At the *semantic level*, we calculate the cosine similarity $\phi$ between sentence embeddings of the original response $R$ and its reconstruction $\hat{R}$ using a pre-trained MiniLM transformer [37], where $\phi \in [-1, 1]$ and $\phi = 1$ indicates perfect semantic alignment.

We define Attack Success Rate (ASR) as the fraction of samples with $\phi > 0.5$, following the empirically validated threshold from prior human-subject studies [38]. These studies show that reconstructions at this similarity level allow human evaluators to reliably identify sensitive topics and user intent, indicating meaningful privacy leakage.

**Implementation Details.** All experiments are implemented in PyTorch, using *Qwen3-8B* as the base reconstruction model and trained with mixed-precision bfloat16. We adopt a parameter-efficient fine-tuning setup with LoRA applied to the attention and feed-forward layers. The model is trained in two stages with stage-specific learning rates to support deterministic alignment learning and merged-token reasoning adaptation. Inference is performed using length-constrained decoding with beam search to promote stable reconstructions. All experiments are conducted on a dedicated server equipped with eight NVIDIA A100 GPUs with 80 GB of memory each. Detailed training configurations and hyperparameters are provided in Appendix C.

## 5.2 Overall Attack Performance

We evaluate the end-to-end reconstruction accuracy of *PromptEcho* on both a constructed single-token UltraChat-derived dataset and two real-world merged-token corpora collected from GPT-4o and DeepSeek-V3. In all experiments, PromptEcho operates under the passive threat model described in Section 3.1, using captured merged-token length sequences as inputs.

Table 3 summarizes the quantitative results. We prioritize reconstruction of the first response segment, which typically conveys the primary topic and user intent and therefore represents the most privacy-sensitive portion of the response. Each subsequent row extends the evaluation to longer response prefixes by including additional segments, capturing how reconstruction fidelity and practical privacy risk evolve as longer responses are considered.

**Single-token Scenario (GPT-4 Turbo).** We first evaluate *PromptEcho* in the single-token setting using a dataset derived from GPT-4 Turbo responses. This setting corresponds to a deterministic alignment scenario, where each observed length maps to a single generated token and provides the least ambiguous reconstruction signal. As shown in Table 3, *PromptEcho* achieves its highest reconstruction accuracy under this setting. For the first segment, which typically conveys the main topic of the response, the attack attains an ASR of 57.66%, with 34.81% of samples reaching $\phi > 0.9$, indicating near-perfect semantic recovery. When considering the full response, overall ASR remains at 40.87%, and ROUGE and ED metrics show consistent reconstruction quality. Despite operating under the most favorable conditions, exact reconstruction becomes increasingly difficult for longer responses under strict evaluation criteria. Nevertheless, the recovered text remains semantically faithful, revealing core intent even when exact character- or token-level matches are imperfect. This single-token setting therefore establishes a strong structural prior and serves as a reference condition under minimal ambiguity, informing effective adaptation to merged-token reconstruction in real-world encrypted deployments.

**Merged-token Scenario (GPT-4o and DeepSeek-V3).** We evaluate *PromptEcho* on real-world merged-token datasets collected from API-based interactions with GPT-4o and DeepSeek-V3. As expected, reconstruction accuracy degrades compared to the single-token reference as multiple tokens are aggregated into each transmitted unit. Nevertheless, *PromptEcho* maintains meaningful recovery capability in both settings. For the first segment, ASR exceeds 42.50% on GPT-4o and 35.65% on DeepSeek-V3, with a substantial fraction of samples achieving high ROUGE scores and low edit distance, indicating that the reconstructed text preserves both character-level, token-level fidelity and semantic content. As more segments are included, ASR ranges from 26.35% and 42.50% for GPT-4o and 20.58% and 35.65% for DeepSeek-V3. The reconstruction accuracy decreases smoothly across all metrics, reflecting the cumulative uncertainty introduced by token aggregation and increasing sequence length rather than abrupt reconstruction failure. Furthermore, we observe that reconstruction accuracy also varies notably across response types. Shorter, topic-focused responses with clear sentence structure tend to yield higher ASR, while longer, list-style or multi-topic responses introduce greater ambiguity accumulation and lower fidelity. A detailed breakdown by prompt type and response length is provided in Section 5.3.

We further observe that GPT-4o consistently outperforms DeepSeek-V3 in reconstruction accuracy. This gap may be explained by two factors. First, DeepSeek-V3 responses are typically longer, which amplifies ambiguity and error propagation over successive segments. Second, our single-token alignment stage is trained on GPT-style data, which likely bet-

Table 3: Overall attack performance on synthetic and real-world datasets. Each row indicates how many leading segments are considered. $N_s$ denotes the average number of output tokens. Results are reported in percent.

| Victim LLM | Segments | $N_s$ | ASR | ϕ>0.9 | ϕ=1.0 | R1>0.9 | R1=1.0 | RL>0.9 | RL=1.0 | ED<=0.1 | ED=0.0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **GPT-4 Turbo (Single Token)** | **1** | 18.38 | **57.66** | 34.81 | 20.31 | 36.21 | 20.31 | 35.69 | 24.78 | 36.69 | 19.75 |
| | **2** | 37.15 | 46.93 | 28.33 | 12.53 | 29.47 | 12.53 | 29.05 | 15.28 | 29.86 | 12.18 |
| | **3** | 54.12 | 48.09 | 29.03 | 10.91 | 30.20 | 10.91 | 29.76 | 13.32 | 30.60 | 10.61 |
| | **5** | 86.30 | 45.10 | 27.23 | 8.35 | 28.32 | 8.35 | 27.92 | 10.18 | 28.70 | 8.12 |
| | **10** | 156.22 | 41.35 | 24.96 | 5.80 | 25.97 | 5.80 | 25.59 | 7.07 | 26.31 | 5.64 |
| | **all** | 220.22 | 40.87 | 24.67 | 4.67 | 25.66 | 4.67 | 25.30 | 5.69 | 26.01 | 4.54 |
| **GPT-4o (Merged Token)** | **1** | 26.23 | **42.50** | 25.50 | 14.87 | 26.14 | 14.79 | 25.86 | 17.75 | 26.36 | 14.06 |
| | **2** | 50.81 | 35.35 | 21.21 | 9.71 | 22.50 | 9.72 | 22.16 | 11.66 | 22.66 | 9.23 |
| | **3** | 78.07 | 32.95 | 19.77 | 7.85 | 20.32 | 7.85 | 20.15 | 9.42 | 20.65 | 7.46 |
| | **5** | 120.32 | 30.90 | 18.54 | 6.16 | 19.11 | 6.16 | 18.62 | 7.39 | 19.12 | 5.85 |
| | **10** | 222.21 | 27.52 | 16.51 | 4.30 | 17.82 | 4.06 | 17.45 | 4.87 | 17.95 | 3.86 |
| | **all** | 303.31 | 26.35 | 15.81 | 3.66 | 16.75 | 3.66 | 16.64 | 4.39 | 17.14 | 3.48 |
| **DeepSeek-V3 (Merged Token)** | **1** | 31.53 | **35.65** | 21.39 | 12.48 | 22.80 | 12.58 | 22.57 | 15.09 | 23.07 | 11.95 |
| | **2** | 63.80 | 28.72 | 17.23 | 7.89 | 18.25 | 7.89 | 18.08 | 9.46 | 18.58 | 7.49 |
| | **3** | 88.30 | 26.44 | 15.86 | 6.30 | 17.33 | 6.30 | 16.86 | 7.56 | 17.36 | 5.98 |
| | **5** | 146.59 | 24.34 | 14.60 | 4.85 | 16.00 | 4.85 | 15.53 | 5.82 | 16.03 | 4.61 |
| | **10** | 254.76 | 22.30 | 13.38 | 3.49 | 13.97 | 3.21 | 13.74 | 3.86 | 14.24 | 3.05 |
| | **all** | 424.82 | 20.58 | 12.35 | 2.86 | 13.24 | 3.06 | 12.99 | 3.67 | 13.49 | 2.90 |

ter matches the lexical and formatting priors of GPT-family outputs, thereby improving transfer to GPT-4o in the merged-token setting. From a privacy perspective, even the lower ASR observed on DeepSeek-V3 (20.58%-35.65%) indicates that a substantial fraction of responses leak semantically identifiable content, particularly in the first segment where core user intent is most concentrated.

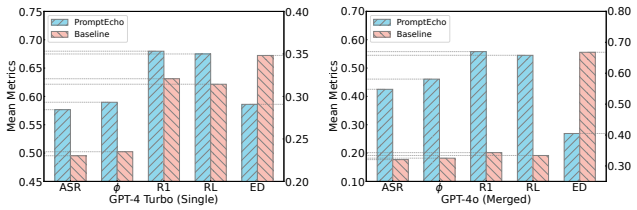
Overall, these results demonstrate that even under encrypted and merged-token transmission, structural token-length patterns leak sufficient information to enable semantically consistent reconstruction of response content, suggesting meaningful privacy leakage is achievable under realistic encrypted deployment conditions.

**Ambiguity Accumulation and Failure Modes.** Although *PromptEcho* achieves meaningful semantic reconstruction under merged-token transmission, reconstruction fidelity gradually decreases as additional segments are incorporated. As Table 3 shows, ASR drops from 42.5% on the first segment to 35.5% and 32.95% when evaluated over two and three segments on GPT-4o. Since PromptEcho conditions later segments on preceding merged token-length sequences as context rather than reconstructed text, the degradation is primarily driven by ambiguity accumulation instead of context error propagation. As response length increases, the number of plausible token combinations grows exponentially, progressively enlarging the candidate reconstruction space. Moreover, compared to the initial segment, later segments tend to contain supporting details and stylistic continuations with weaker topic-focused semantic signals, further reducing reconstruction precision.

Beyond response length, reconstruction failures are also strongly affected by the intrinsic linguistic characteristics of the generated output. At the lexical level, rare terminology and highly diverse phrasing patterns weaken semantic alignment cues and increase reconstruction difficulty. At the structural level, outputs governed by strict syntactic or logical constraints such as code generation and mathematical reasoning are highly sensitive to small reconstruction errors. At the semantic level, multi-topic and list-style responses introduce additional ambiguity, as multiple semantically plausible candidate reconstructions remain equally consistent with merged token-length observations alone. Consequently, outputs with low semantic redundancy, as well as responses containing many numerals, code fragments, or entity-heavy content, are among the most challenging cases for reconstruction. For natural-language responses, reconstruction degradation is typically gradual, not catastrophic, with semantic consistency remaining partially preserved due to persistent discourse-level linguistic regularities despite lexical imprecision.

In contrast, shorter topic-focused responses with stable discourse structures generally provide stronger semantic alignment cues and higher reconstruction fidelity. Representative reconstruction examples are discussed in Section 5.7.

**Comparing with Existing Methods.** We further compare




(a) GPT-4 Turbo (Single) (b) GPT-4o (Merged)

Figure 4: Reconstruction comparison of *PromptEcho* v.s. baseline across different transmission scenarios.

*PromptEcho* with a prior method designed for single-token transmission [38]. The comparison is conducted on both the single-token dataset and the merged-token GPT-4o dataset and shown in Figure 4. In the single-token setting, *PromptEcho* consistently outperforms the baseline, achieving nearly 8% improvement in ASR and a 10% increase in high-fidelity reconstructions (e.g., $\phi > 0.9$ and exact-match metrics). These results demonstrate that integrating reasoning-guided reconstruction with unified token-length representation significantly enhances recovery precision, even under deterministic conditions. Conversely, the prior method suffers a severe performance substantial degradation in the merged-token setting, with semantic metrics dropping to near 0.2 and ED exceeding 0.6 due to a rigid reliance on explicit token boundaries and one-to-one length alignment. Meanwhile, by leveraging semantic-aware segmentation and transferring deterministic linguistic regularities, *PromptEcho* maintains robust reconstruction capabilities in merged-token transmission. This disparity underscores that while traditional attacks fail to generalize to merged-token API deployments, PromptEcho semantics-aware design effectively navigates the uncertainty inherent in merged-token leakage.

## 5.3 Attack Scope and Domain Generalization

Building on the overall results, this section examines how reconstruction performance varies across semantic domains and prompt types, and whether the attack generalizes to topics unseen during training. These analyses characterize the conditions under which the attack succeeds or fails, directly informing the practical scope of the privacy risk.

**Topic-level Privacy Leakage.** To assess the semantic scope of the reconstruction attack, we analyze topic-level privacy leakage from the first segment across three datasets spanning both single-token and merged-token transmission scenarios. Using GPT-4o as an automatic annotator, reconstructed sentences are categorized by subject domain and privacy sensitivity (e.g., health, personal identity, life status, relationships, and education). Figure 5 reports the ASR for each topic across datasets. Overall, privacy leakage is relatively uniform across topics, indicating that the reconstruction attack can expose semantically identifiable information across a broad range of topics and does not exhibit a strong bias to any specific subject domain. Slightly higher leakage is observed in daily-life and personal topics, where responses tend to follow common conversational patterns and exhibit stronger linguistic regularities. These results suggest that privacy leakage from merged token-length side channels is not confined to a narrow set of sensitive domains, but instead reflects a general vulnerability affecting a wide range of content. Even responses to benign prompts may reveal user intent and contextual information when coherent semantics information can be inferred from encrypted traffic. This effect is largely driven by the base model's broad semantic knowledge and the diverse, topically balanced training data, which jointly mitigate topic-specific bias in the reconstruction priors.

**Reconstruction Across Prompt Types.** To further characterize when reconstruction succeeds or fails, we evaluate PromptEcho across several representative prompt types exhibiting different response structures and complexity levels, including general question answering, long-form explanation, multi-topic responses, code generation, mathematical reasoning, and creative writing. To construct this evaluation set, we prompt ChatGPT to generate representative questions for each prompt type while controlling for comparable response length distributions. Table 4 summarizes the average ASR under each prompt type interaction with GPT-4o. We observe that natural-language QA achieves the highest ASR 41.2%, as responses typically follow common conversational patterns with strong semantic regularity, stable discourse patterns and concentrated semantic content. In contrast, reconstruction performance decreases for long-form explanations at 29.5% and multi-topic responses at 32.6%, where ambiguity accumulates across longer outputs and multiple semantically plausible candidates may satisfy similar merged token-length observations. Creative-writing prompts also exhibit lower ASR at 22.8%, reflecting the increased diversity and variability of generated phrasing patterns. Structurally constrained outputs such as code generation and mathematical reasoning become substantially more difficult at nearly 10% ASR, as small token-level reconstruction errors can significantly alter syntactic validity or logical correctness. Overall, the results suggest that reconstruction effectiveness is more strongly associated with response complexity, semantic concentration, and structural regularity than with prompt topic alone, with structurally constrained and stylistically diverse outputs representing the most challenging reconstruction cases.

**Generalization to Unseen Domains.** Having established that privacy exposure is prevalent across topics, we now investigate whether the attack generalizes to new semantic domains unseen during training. While the UltraChat-derived dataset covers a broad range of conversational topics, its questions mainly focus on general inquiries, with limited representation of specialized or emerging topics. To simulate unseen domains, we prompt GPT-4o to generate questions from creative writing, law, and niche scientific fields, forming an OOD

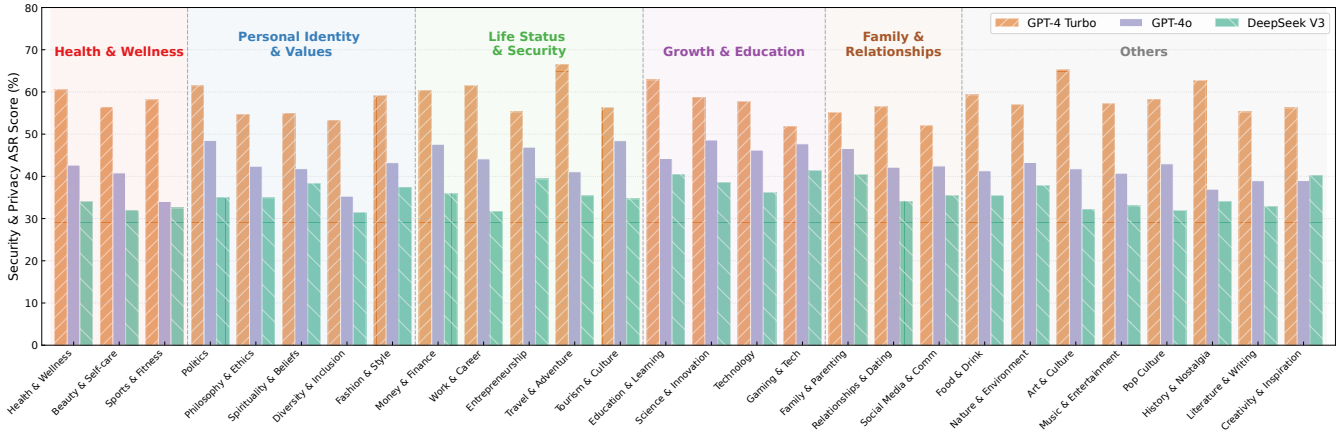
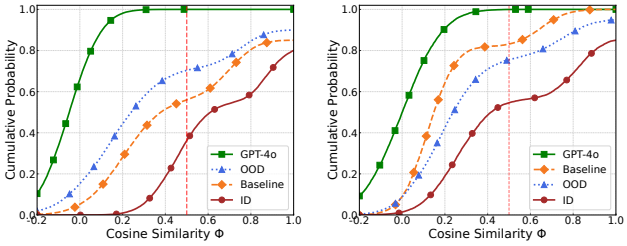



Figure 5: Topic-level reconstruction success across single-token and merged-token scenarios.

Table 4: Reconstruction performance in GPT-4o across different prompt types.

| Prompt Type | Example | ASR |
|---|---|---|
| General QA | Daily-life questions | 41.2 |
| Multi-topic / List-style | Enumerations & comparisons | 32.6 |
| Explanatory requests | Detailed explanations | 29.5 |
| Creative Writing | Poetry creation | 22.8 |
| Code Generation | Python / SQL generation | 11.5 |
| Mathematical Reasoning | Formula derivation | 10.9 |




(a) GPT-4 Turbo (b) GPT-4o

Figure 6: Cumulative cosine similarity comparison.

evaluation set. As shown in Figure 6, ASR drops from 57.7% on in-distribution (ID) samples to 32.6% on OOD topics in single transmission, and from 42.5% to 22.3% in GPT-4o, reflecting reduced coverage of domain-specific linguistic patterns. However, nearly 33% and 20% of OOD reconstructions still retain sufficient semantic content for topic inference in single and merged transmission, suggesting partial transfer of structural priors to new domains. The performance drop is consistent with the lexical-level failure mode identified above, OOD reconstructions miss domain-specific terms or specific details that fall outside learned alignment priors, while high-level discourse structures, such as question–answer or opinion–justification patterns, remain partially recoverable. This indicates that *PromptEcho* relies on broad linguistic patterns learned during training rather than memorizing specific topic knowledge. Accordingly, defenses based solely on limiting training topic coverage are unlikely to be sufficient, effective mitigations must target the model's ability to infer linguistic structure from encrypted traffic, as discussed in Section 6.

## 5.4 Robustness Analysis

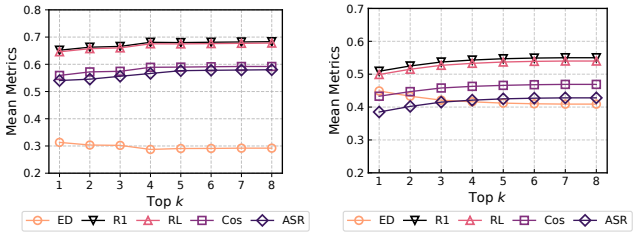
Previous sections evaluate *PromptEcho* under controlled settings where merged-token sequences are fully observed and correctly aligned. We now examine its robustness under imperfect network conditions commonly encountered in real-world API traffic. Based on traffic collected from GPT-4o and DeepSeek-V3 interactions, we identify several non-ideal factors in encrypted LLM communication and systematically evaluate their impact.

**Preamble Interference.** In many sessions, variable-length preambles appear before the actual response stream. Although API protocols typically use a fixed-length header, connection reuse or additional service fields can introduce variability in the preamble length. This change obscures the initial token boundary, preventing reliable separation of header bytes from the first merged token based solely on packet length. To evaluate this effect, we remove the first merged token from each sequence and retrain the model on truncated inputs. The resulting ASR decreases by only 2–3%, as the initial token typically carries limited semantic information. As shown in Table 5, *PromptEcho* can still recover coherent responses in different API interactions when preamble interference.

**Incomplete or Truncated.** Real-world captures may miss packets due to network interruption or limited capture windows, leading to responses end truncated or occasional gaps. We simulate this condition by removing the final several merged tokens or randomly dropping short segments within

Table 5: Robustness analysis under imperfect conditions.

| Condition | Scenario | ASR | ϕ | R1 | RL | ED |
|---|---|---|---|---|---|---|
| **Preamble** | GPT-4 Turbo | 55.34 | 57.08 | 66.52 | 65.96 | 30.32 |
| | GPT-4o | 40.13 | 44.60 | 52.46 | 51.63 | 43.49 |
| | DeepSeek-v3 | 34.20 | 39.95 | 47.21 | 46.18 | 48.51 |
| **Incomplete & Truncated** | GPT-4 Turbo | 53.10 | 55.07 | 64.18 | 63.37 | 32.53 |
| | GPT-4o | 37.28 | 43.25 | 50.84 | 49.12 | 44.15 |
| | DeepSeek-v3 | 32.50 | 37.97 | 44.87 | 43.89 | 50.28 |




(a) GPT-4 Turbo (Single) (b) GPT-4o (Merged)

Figure 7: Effect of Top-$k$ selection on reconstruction performance.

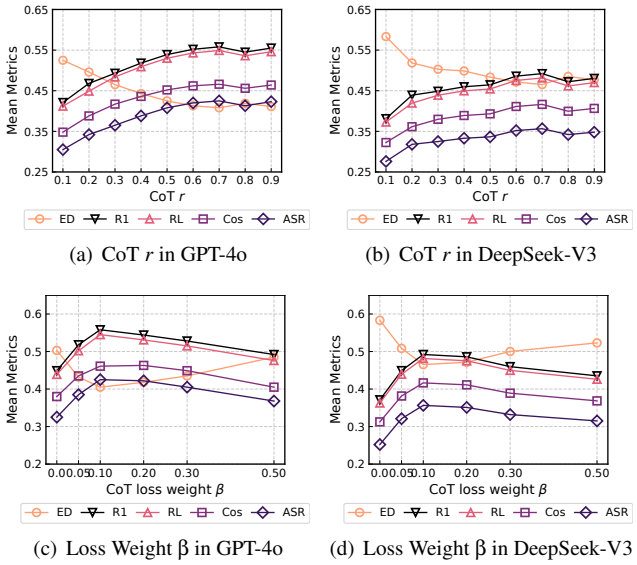
the sequence. Overall ASR decreases moderately (by 4–6%), while reconstructed outputs remain semantically recognizable. Missing tokens near the end of a response have a negligible impact, as they typically correspond to formulaic closing phrases with weak topic relevance (e.g., polite completions or acknowledgments). Mid-sequence losses may disrupt local syntax but rarely alter the overall semantics. These results confirm that privacy exposure is primarily driven by the recovery of first segments, which usually contain the main topic or user intent.

Across all perturbations, *PromptEcho* remains effective under imperfect observation conditions. The reasoning-guided reconstruction mitigates segmentation noise, while progressive training preserves linguistic grounding despite partial or noisy inputs. These results show that merged-token length inference attacks remain viable in practical deployments, underscoring the security risk posed by token-length side channels in encrypted LLM API traffic.

## 5.5 Parameter Analysis

We analyze the impact of key hyperparameters of different stages of our attack pipeline. Specifically, we examine inference-time parameters that govern candidate selection and ranking, as well as training-time parameters that control the strength and proportion of reasoning-based (CoT) supervision used for merged-token decomposition.

**Top-$k$ Selection.** During inference, each segment yields multiple candidate reconstructions, which are ranked using a composite score that combines length consistency and generation likelihood. The highest-ranked candidate is selected as the final reconstruction $\hat{R}_x$. To evaluate the effectiveness of this ranking strategy, we vary the top-$k$ threshold and report the performance assuming the correct reconstruction appears within the top-$k$ candidates. As shown in Figure 7, reconstruction performance improves rapidly with increasing $k$ and closely approaches the upper bound at $k = 5$. Further increasing $k$ yields only marginal gains, indicating diminishing returns. These results suggest the scoring function effectively promotes high-fidelity candidates to the top of the ranking and that a small beam size is sufficient for reliable reconstruction.



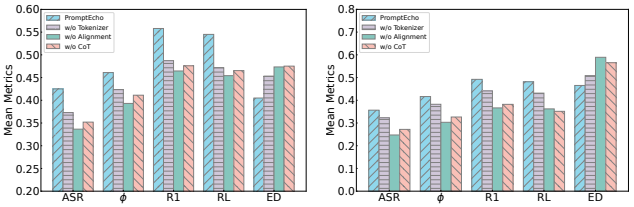

(a) CoT $r$ in GPT-4o (b) CoT $r$ in DeepSeek-V3

(c) Loss Weight β in GPT-4o (d) Loss Weight β in DeepSeek-V3

Figure 8: Effect of CoT ratio $r$ and loss weight β on reconstruction performance.

**CoT Ratio $r$.** We next vary the proportion $r$ of training samples that include explicit CoT supervision. Figure 8 shows that introducing CoT reasoning substantially improves performance in merged-token reconstruction, particularly for longer sequence datasets like DeepSeek-v3. However, the improvement saturates when $r > 0.75$, as excessive reasoning traces can introduce redundant intermediate tokens and slow convergence. An optimal trade-off is achieved when $r$ is between 0.5 and 0.75, balancing explicit decomposition guidance and generalization efficiency.

**CoT Loss Weight β.** We further analyze the sensitivity of the weighting coefficient β used in the reasoning-based decomposition objective. This parameter controls the relative strength of supervision on the reasoning segment compared with the final answer segment. To examine its effect, we fine-tuned the model with different β values $\{0.05, 0.1, 0.2, 0.3, 0.5\}$. As shown in Figure 8, reconstruction accuracy improves as β rises from 0 to 0.2 and peaks between 0.1 and 0.2, where the model achieves the highest ASR and R1 scores. This indicating that moderate reasoning supervision helps the model




(a) GPT-4o (Merged) (b) DeepSeek-V3 (Merged)

Figure 9: Ablation study of key components in *PromptEcho*.

learn more reliable token decomposition. When $\beta = 0$, i.e., without reasoning supervision, the model overfits local token patterns and struggles with longer merged segments. In contrast, when $\beta$ exceeds 0.3, performance slightly declines due to the model overemphasizing reasoning trace generation, which introduces verbosity and weakens final reconstruction coherence. Overall, a moderate $\beta$ (0.1–0.2) provides the best balance, strengthening structural inference without compromising linguistic fluency.

## 5.6 Ablation Study

To assess the contribution of individual components in *PromptEcho*, we conduct an ablation study and summarize the results in Figure 9.

**Unified Tokenizer.** We replace the custom symbolic tokenizer with raw numeric length inputs, removing the distinction between single-token symbols $s_n$ and merged-token symbols $M_x$. This change causes a consistent performance drop across all metrics, with ASR decreasing by over 5%, and both ROUGE and edit-distance scores degrade substantially. Without symbolic alignment, the model treats token lengths as unstructured numeric values in word-level, leading to broken syntax, repetition, and loss of semantic coherence. These results show that the tokenizer is essential for aligning length observations with natural-language representations.

**Deterministic Alignment Finetuning.** We remove single-token length alignment fine-tuning and directly train the base model on merged-token sequences. This leads to a clear degradation in reconstruction quality, with ASR decreases by 9%, and the generated text becomes fragmented with missing context and increased topic drift, especially for longer sequence datasets like DeepSeek-v3. Without deterministic single-token length alignment, the model lacks a stable structural prior, and CoT supervision alone cannot compensate for this missing foundation. In addition, directly training on the smaller merged-token dataset increases overfitting and limits generalization. This highlights the need for progressive fine-tuning: aligning single tokens first establishes transferable linguistic structure for merged-token settings.

**CoT-guided Reasoning.** We disable CoT supervision by removing the think part and setting the corresponding loss weight $\beta = 0$. This results in a noticeable decline in semantic coherence and structural accuracy: ASR drops by 7%, and ROUGE-L scores decrease markedly for merged-token sequences with $N_s \geq 5$. Without explicit reasoning guidance, the model often misaligns token boundaries and produces locally plausible but globally inconsistent text. In contrast, CoT supervision enables the model to infer plausible token decompositions under length constraints, leading to smoother transitions and improved topic consistency across segments.

Overall, the ablation results demonstrate that each component of *PromptEcho* plays a distinct and complementary role. The tokenizer provides representational alignment between length sequences and linguistic units, deterministic alignment fine-tuning establishes a transferable structural prior, and CoT-guided reasoning applies semantic-aware decomposition to resolve ambiguity introduced by token aggregation. Together, these components enable robust and semantically faithful reconstruction from merged token-length side channels.

## 5.7 Case Study of Practical Privacy Leakage

Figure 10 presents representative reconstruction examples spanning multiple leakage levels, including verbatim recovery, partial entity leakage, semantic leakage, structural leakage, and reconstruction failure. The cases cover several sensitive domains such as mental health, cancer treatment, political activism, geopolitical conflicts, and legal discussions.

Examples 1–3 demonstrate that meaningful privacy leakage can occur even when reconstruction is imperfect. Example 1 shows a worst-case verbatim recovery scenario in which the attacker exactly reconstructs a response related to PTSD support groups, directly exposing a highly sensitive mental-health inquiry. Example 2 illustrates partial entity leakage: although several descriptive tokens are paraphrased, medically sensitive entities such as “immunotherapy”, “immune system”, and “cancer cells” are still recovered correctly, allowing the attacker to infer the treatment context. Example 3 further shows that topic-level privacy leakage may persist despite entity corruption. Although “Vancouver” is incorrectly reconstructed as “Australia”, the attacker can still infer that the response concerns Indigenous communities and political activism more broadly.

Examples 4–6 highlight important reconstruction failure modes and practical attack boundaries. In Example 4, the specific geopolitical entity is corrupted (“Venezuela” $\rightarrow$ “Palestine”), while the broader “political and economic crisis” framing remains preserved, illustrating that discourse-level semantic structure can survive substantial lexical errors. Example 5 demonstrates a common template-recovery failure mode: the generic LLM disclaimer structure is reconstructed correctly, leading to moderate lexical overlap despite complete loss of the sensitive payload related to gun-control debates. Finally, Example 6 shows a near-total semantic failure where historically specific and low-frequency entities such as “1967 Sexual Offences Act” are not recovered at all. Although the

high-level causal sentence structure remains partially preserved, both the year token and the legally sensitive topic are corrupted, demonstrating that numerals, rare entities, and historically specific $n$-grams are particularly challenging under merged-token ambiguity.

Overall, the case studies suggest that privacy leakage exists along a continuum ranging from exact verbatim reconstruction to partial semantic inference. Even when token-level recovery is inaccurate, attackers may still infer sensitive medical, political, or legal topics from partially preserved entities or discourse structure. At the same time, the failure cases reveal important practical limitations of the attack, particularly for responses containing numerals, low-frequency entities, or semantically diverse content.

## 6 Discussion

**Potential Mitigations and Trade-offs.** Mitigating token-length side channels may require application-layer defenses such as padding, randomized batching, or aggregating larger portions of the response. However, simply applying these approaches introduces non-trivial trade-offs. Padding increases bandwidth overhead, aggressive aggregation adds latency, and transmitting full responses at once undermines the real-time streaming experience critical for modern LLM services. Moreover, deploying such defenses consistently across diverse protocols and clients is challenging in practice.

A possible solution is dynamic padding, where padding is applied only when token lengths fall below a certain threshold to reduce bandwidth overhead. For instance, naive fixed padding to the Maximum Segment Size increases bandwidth overhead by approximately 3.2× on average, while dynamic padding reduces this to approximately 1.4× while still increasing reconstruction ambiguity. Alternatively, randomizing padding amounts could make the side-channel less predictable. Designing practical defenses that balance efficiency, usability, and privacy remains an open challenge. Future work could explore adaptive techniques tailored to specific patterns, dynamically adjusting defenses based on data sensitivity.

We have reported this side-channel leakage vulnerability with suggested defense solutions to major LLM providers, including OpenAI and DeepSeek. Notably, OpenAI has since deployed a default-on obfuscation mechanism via the `include_obfuscation` parameter in its Responses API [31], opting for dynamic obfuscation to balance privacy and bandwidth efficiency. This industry response further suggests that token-length side channels constitute a practical concern in real-world LLM streaming systems.

**Limitations and Scope.** Our study has several limitations. We focus on single-turn English-language API interactions and passive network adversaries, and do not claim perfect reconstruction of complete outputs. Particularly, the reconstruction exhibits a relatively low success rate in out-of-distribution and

**Attacks on GPT-4o and DeepSeek-V3**

**ϕ: 1.00 R1: 1.00 ED: 0.00 Verbatim Reconstruction**

Here are some resources and support groups for individuals with post-traumatic stress disorder (PTSD).
Here are some resources and support groups for individuals with post-traumatic stress disorder (PTSD).
Privacy Exposure: **Explicit mental-health inquiry**

**ϕ: 0.871 R1: 0.87 ED: 0.27 Partial Entity Leakage**

Immunotherapy: This **breakthrough** involves **training** the body's immune system to attack cancer cells.
Immunotherapy: This **experimental treatment** involves the body's immune system to attack cancer cells.
Privacy Exposure: **Treatment topic disclosed despite paraphrasing**

**ϕ: 0.69 R1: 0.82 ED: 0.18 Semantic Leakage**

Indigenous communities have played a significant role in shaping **Vancouver**'s political landscape and social **activism efforts.**
Indigenous communities have played a significant role in shaping **Australia**'s political landscape and social **identity through,**
Privacy Exposure: **Sensitive political topic inferred despite entity substitution**

**ϕ: 0.46 R1: 0.61 ED: 0.39 Structural leakage**

The political and economic crisis in **Venezuela** has had a severe impact on **its people and neighboring countries**.
The political and economic status of **Palestine** has had a severe impact on **the social and educational landscape**.
Privacy Exposure: **Geopolitical crisis framing preserved despite entity corruption**

**ϕ: 0.15 R1: 0.60 ED: 0.37 Template-only leakage**

I do not possess personal opinions and views, but here are some of the **perspectives of different media outlets on the gun control debate** in the **united states**:
I do not possess personal opinions and views, but here are some of the **advancements in renewable water sources as per the current trends** in the **energy sector**:
Privacy Exposure: **Template recovered but payload lost**

**ϕ: 0.08 R1: 0.39 ED: 0.64 Reconstruction failure**

The **1967 Sexual Offences Act** had a significant impact on the **legal system in England by decriminalizing homosexual acts in private between consenting adults over the age of**
The **2011 floods and cyclones** had a significant impact on the **daily lives of farmers in the region both positively and by causing several challenges during post-disaster.**
Privacy Exposure: **High-level causal framing preserved while sensitive legal event is lost**

Figure 10: Case studies of successful and failed reconstructions across different leakage levels. Each block displays the original model response (top) and the attacker-reconstructed output (bottom), annotated with leakage category, semantic similarity ($\phi$), lexical metrics ($R1, ED$), and privacy exposure.

long-response scenarios, especially when the text contains rare terminology, dense numerals, or complex code fragments. Instead, we demonstrate that partial semantic recovery can reveal measurable semantic information and raise practical privacy risks under our evaluated metrics and threat model. Additionally, our analysis targets API-based deployments and may not directly generalize to all interaction modalities or future protocol designs. Despite these limitations, the observed leakage is rooted in the relationship between response generation and streaming behavior in the evaluated API settings. Similar risks may exist in other remote LLM services that adopt comparable streaming mechanisms, which remains an important direction for future study. Exploring defenses that reduce such leakage while maintaining efficiency remains an important direction for future work.

**Timing Side-Channel and Future Directions.** This work intentionally focuses on packet-size leakage derived from TLS record sizes, which provides stable observables for merged token-length inference. Although timing-based side channels may provide additional information for segmentation and inference, they are highly sensitive to network jitter, server-side scheduling, and client-side variability, making them less reliable in real-world deployments. In preliminary measurements, repeated queries of a fixed phrase exhibited substantial timing variability across runs. We therefore isolate length-based leakage to establish a robust lower-bound privacy leakage attack without relying on timing information. Nevertheless, timing signals may still contain complementary information about token generation behavior. Exploring multi-signal fusion strategies that incorporate timing information, as well as characterizing attack performance under varying concurrency levels or multi-turn workloads, remain important directions for future work, both for understanding stronger inference attacks and for designing more effective defenses.

**Generalization across Services and Base Models.** The empirical results from OpenAI GPT-4o and DeepSeek demonstrate the transferability of *PromptEcho* across different API services. As shown in Table 1, we successfully attack several services even without targeted data training. This effectiveness is driven by the structural consistency of LLM responses, where models from various platforms share common linguistic patterns and sentence structures. These findings indicate that the observed privacy risks are not specific to any single provider but reflect a broader vulnerability in the LLM API ecosystem. Furthermore, *PromptEcho* remains agnostic to the underlying base models. Adapting to a new model involves adjusting the tokenizer and retraining to align the semantic space, ensuring the effectiveness regardless of the adversary's model architecture. This flexibility allows our approach to remain potent across different service providers and backbone configurations, highlighting a universal security challenge that requires more robust defenses in encrypted API deployments.

## 7 Related Work

Unlike most existing privacy attacks targeting LLMs' training data [26, 33, 39], system prompt [24, 44] or model parameters [9, 22, 45], our work focuses on privacy leakage during the inference phase, specifically sensitive user information revealed during interactions with the model. Existing research can be broadly categorized into two types: active and passive.

**Active Attacks.** Active attacks require the adversary to share the same target model as the victim to trigger crafted requests and observe the model's behavior, such as the response text [10] or timing-based side channels [35, 47], to infer sensitive information. However, these attacks typically assume an overly strong threat model (e.g., stable execution without network noise or cache-related interference) and require direct probing, which can expose malicious queries to the victim service, thereby undermining stealth and practicality in real-world settings.

**Passive Attacks.** In contrast, passive attacks exploit side-channel information obtained through network services or software vulnerabilities without requiring direct interaction with the target model or the ability to inject prompts. Depending on the deployment setting, passive attacks can be divided into targeting locally deployed LLMs and remote LLM services. Recent work has revealed that Local LLMs are vulnerable to hardware-based side-channel attacks [1, 18], where adversaries monitor CPU cache access patterns or exploit timing variations in autoregressive decoding to infer token values and positions. However, these attacks are limited to shared CPU caches and are ineffective when token embedding is performed on GPUs, requiring prior knowledge of the model's memory layout and embedding table structure.

Remote LLM services, on the other hand, are typically targeted by attacks leveraging network-level side channels, inspired by traditional tasks like website fingerprinting [21] and malicious traffic detection [17]. Packet arrival timing-based attacks [8, 46] infer the number of tokens generated by the model based on the timing of response packets, while packet-length-based keylogging attacks [38] attempt to reconstruct model outputs by analyzing the sequence of packet sizes. However, timing-based attacks are hindered by network latency fluctuations, while the merging of multiple tokens into a single packet in modern transmission mechanisms makes attacks based on single-token length difficult to extract accurate token-level leakage, limiting the feasibility of these attacks in real-world deployment environments.

## 8 Conclusion

Modern LLM API services that employ merged-token streaming inadvertently introduce a novel merged-token length side channel: even under encryption, observable length patterns can leak substantive information about LLMs' response con-

tent. We demonstrate the practical feasibility of exploiting this leakage with PromptEcho, a generative eavesdropping attack that frames inference as a length-constrained sequence reconstruction problem. By combining semantics-aware decomposition with linguistic priors, PromptEcho reduces merged-length ambiguity and enables reconstruction of semantically consistent responses from encrypted streams. Overall, our findings highlight an important and previously underexplored privacy risk in the evaluated API-based LLM settings, which suggests that fine-grained response content may expose unintended information through observable traffic patterns, even when traffic is encrypted. We hope these findings motivate the development of stronger and more practical protections that mitigate or decorrelate length-based leakage in streaming LLM deployments.

## Acknowledgments

We are grateful to our shepherd and the anonymous reviewers for their valuable guidance and insightful comments. This work is supported in part by the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China under Grant JYB2025XDXM114, the National Natural Science Foundation of China under Grants 62132011 and 62425201, and the National Cyber Security–National Science and Technology Major Project under Grant 2025ZD1503600. Qi Li is the corresponding author.

## Ethical Considerations

This work investigates privacy risks in API-based LLM services by analyzing side-channel leakage from encrypted traffic. Our study is conducted solely to understand and mitigate such vulnerabilities, not to enable exploitation. Relevant stakeholders include end users of LLM services, organizations deploying LLM APIs in sensitive settings, API providers, and downstream application developers.

Our experiments rely on two data sources: (i) publicly available datasets and (ii) a self-collected dataset of encrypted API interaction traces obtained from our own controlled sessions with publicly accessible LLM APIs. We do not capture, intercept, or analyze communications or traffic associated with real users. Moreover, all datasets used in this study do not contain any harmful content. We do not access decrypted payloads, internal model states, or proprietary service infrastructure beyond what a passive on-path observer could infer from encrypted metadata by design.

We believe that responsibly disclosing these risks is important for improving awareness of emerging side-channel threats, strengthening the security and privacy of deployed LLM systems, and encouraging the development of practical mitigations. Following responsible disclosure practices, we proactively notified the security teams of multiple providers, including OpenAI and DeepSeek, about the merged token-length side-channel vulnerability prior to submission and are actively engaging with them on mitigation. We also note that OpenAI has since deployed dynamic obfuscation to mitigate this type of leakage, suggesting that research of this nature can motivate concrete defensive actions in practice. Our goal is not to facilitate malicious eavesdropping, but to expose a fundamental architectural weakness in encrypted API streaming and motivate stronger defenses. By demonstrating the feasibility of *PromptEcho*, we aim to encourage the development and deployment of effective countermeasures, such as dynamic padding or increasing the token aggregation granularity, to better protect user privacy in the broader LLM ecosystem.

## Open Science

To support the reproducibility of our findings, we release the core implementation and datasets of *PromptEcho*. The artifact includes: (1) the two-stage fine-tuning scripts, (2) the inference and evaluation scripts, (3) the training and test datasets consisting of token-length sequences and corresponding text references, and (4) anonymized exemplar network traces collected from ChatGPT and DeepSeek API sessions. The artifact is archived at: https://doi.org/10.5281/zenodo.20441510

## References

[1] Andrew Adiletta and Berk Sunar. Spill the beans: Exploiting cpu cache side-channels to leak tokens from large language models. *arXiv preprint arXiv:2505.00817*, 2025.

[2] DeepSeek AI. Deepseek chat. https://chat.deepseek.com/.

[3] StepFun AI. Stepfun. https://stepfun.ai.

[4] ZhiPu AI. Glm. https://z.ai.

[5] Alibaba. Qwen. https://tongyi.aliyun.com/.

[6] Alibaba Group. Qwen tokenizer and model files. https://huggingface.co/Qwen.

[7] Anthropic. Anthropic tokenizer. https://www.npmjs.com/package/@anthropic-ai/tokenizer.

[8] Nicholas Carlini and Milad Nasr. Remote timing attacks on efficient language model inference. *arXiv preprint arXiv:2410.17175*, 2024.

[9] Nicholas Carlini, Daniel Paleka, Krishnamurthy Dj Dvijotham, Thomas Steinke, Jonathan Hayase, A Feder Cooper, Katherine Lee, Matthew Jagielski, Milad Nasr,

Arthur Conmy, et al. Stealing part of a production language model. In *International Conference on Machine Learning*, pages 5680–5705. PMLR, 2024.

[10] Junjie Chu, Zeyang Sha, Michael Backes, and Yang Zhang. Reconstruct your previous conversations! comprehensively investigating privacy leakage risks in conversations with gpt models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6584–6600, 2024.

[11] Pierre Colombo, Telmo Pessoa Pires, Malik Boudiaf, Dominic Culver, Rui Melo, Caio Corro, Andre FT Martins, Fabrizio Esposito, Vera Lúcia Raposo, Sofia Morgado, et al. Saullm-7b: A pioneering large language model for law. *arXiv preprint arXiv:2403.03883*, 2024.

[12] Jiaxi Cui, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. Chatlaw: Open-source legal large language model with integrated external knowledge bases. *CoRR*, 2023.

[13] Tianyu Cui, Xinjie Lin, Sijia Li, Miao Chen, Qilei Yin, Qi Li, and Ke Xu. Trafficllm: Enhancing large language models for network traffic analysis with generic traffic representation. *arXiv preprint arXiv:2504.04222*, 2025.

[14] DeepSeek AI. Deepseek model tokenizer. https://huggingface.co/deepseek-ai.

[15] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023.

[16] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.

[17] Chuanpu Fu, Qi Li, Meng Shen, and Ke Xu. Realtime robust malicious traffic detection via frequency domain analysis. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3431–3446, 2021.

[18] Zibo Gao, Junjie Hu, Feng Guo, Yixin Zhang, Yinglong Han, Siyuan Liu, Haiyang Li, and Zhiqiang Lv. I know what you said: Unveiling hardware cache {Side-Channels} in local large language model inference. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 1649–1668, 2025.

[19] Google. Gemini token counting api. https://ai.google.dev/docs/tokens.

[20] Google. Google gemini. https://gemini.google.com/.

[21] Jamie Hayes and George Danezis. k-fingerprinting: A robust scalable website fingerprinting technique. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 1187–1203, 2016.

[22] Eliahu Horwitz, Jonathan Kahana, and Yedid Hoshen. Recovering the pre-fine-tuning weights of generative models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 18882–18904, 2024.

[23] Allen H Huang, Hui Wang, and Yi Yang. Finbert: A large language model for extracting information from financial text. *Contemporary Accounting Research*, 40(2):806–841, 2023.

[24] Bo Hui, Haolin Yuan, Neil Gong, Philippe Burlina, and Yinzhi Cao. Pleak: Prompt leaking attacks against large language model applications. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 3600–3614, 2024.

[25] Xinjie Lin, Gang Xiong, Gaopeng Gou, Zhen Li, Junzheng Shi, and Jing Yu. Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification. In *Proceedings of the ACM Web Conference 2022*, pages 633–642, 2022.

[26] Ruixuan Liu, Tianhao Wang, Yang Cao, and Li Xiong. Precurious: How innocent pre-trained language models turn into privacy traps. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 3511–3524, 2024.

[27] Xiao-Yang Liu, Guoxuan Wang, Hongyang Yang, and Daochen Zha. Fingpt: Democratizing internet-scale data for financial large language models. *arXiv preprint arXiv:2307.10485*, 2023.

[28] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. Deepcorr: Strong flow correlation attacks on tor using deep learning. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 1962–1976, 2018.

[29] OpenAI. Chatgpt. https://chatgpt.com/.

[30] OpenAI. Learn about language model tokenization. https://platform.openai.com/tokenizer.

[31] OpenAI. Openai developers' api reference document. https://developers.openai.com/api/reference/resources/responses/methods/create.

[32] Cheng Peng, Xi Yang, Aokun Chen, Kaleb E Smith, Nima PourNejatian, Anthony B Costa, Cheryl Martin, Mona G Flores, Ying Zhang, Tanja Magoc, et al. A study of generative large language model for medical research and healthcare. *NPJ digital medicine*, 6(1):210, 2023.

[33] Ahmed Salem, Giovanni Cherubin, David Evans, Boris Köpf, Andrew Paverd, Anshuman Suri, Shruti Tople, and Santiago Zanella-Béguelin. Sok: Let the privacy games begin! a unified treatment of data inference privacy in machine learning. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 327–345. IEEE, 2023.

[34] Dong Shu, Haoran Zhao, Xukun Liu, David Demeter, Mengnan Du, and Yongfeng Zhang. Lawllm: Law large language model for the us legal system. In *Proceedings of the 33rd ACM International Conference on information and knowledge management*, pages 4882–4889, 2024.

[35] Linke Song, Zixuan Pang, Wenhao Wang, Zihao Wang, XiaoFeng Wang, Hongbo Chen, Wei Song, Yier Jin, Dan Meng, and Rui Hou. The early bird catches the leak: Unveiling timing side channels in llm serving systems. *IEEE Transactions on Information Forensics and Security*, 2025.

[36] Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature medicine*, 29(8):1930–1940, 2023.

[37] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.

[38] Roy Weiss, Daniel Ayzenshteyn, and Yisroel Mirsky. What was your prompt? a remote keylogging attack on {AI} assistants. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 3367–3384, 2024.

[39] Rui Wen, Zheng Li, Michael Backes, and Yang Zhang. Membership inference attacks against in-context learning. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 3481–3495, 2024.

[40] Charles V Wright, Lucas Ballard, Fabian Monrose, and Gerald M Masson. Language identification of encrypted voip traffic: Alejandra y roberto or alice and bob? In *USENIX Security Symposium*, volume 3, pages 43–54, 2007.

[41] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023.

[42] Diwen Xue, Michalis Kallitsis, Amir Houmansadr, and Roya Ensafi. Fingerprinting obfuscated proxy traffic with encapsulated {TLS} handshakes. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 2689–2706, 2024.

[43] Xi Yang, Aokun Chen, Nima PourNejatian, Hoo Chang Shin, Kaleb E Smith, Christopher Parisien, Colin Compas, Cheryl Martin, Anthony B Costa, Mona G Flores, et al. A large language model for electronic health records. *NPJ digital medicine*, 5(1):194, 2022.

[44] Yong Yang, Changjiang Li, Yi Jiang, Xi Chen, Haoyu Wang, Xuhong Zhang, Zonghui Wang, and Shouling Ji. Prsa: Prompt stealing attacks against large language models. *arXiv preprint arXiv:2402.19200*, 2024.

[45] Boyang Zhang, Xinlei He, Yun Shen, Tianhao Wang, and Yang Zhang. A plot is worth a thousand words: model information stealing attacks via scientific plots. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 5289–5306, 2023.

[46] Tianchen Zhang, Gururaj Saileshwar, and David Lie. Time will tell: Timing side channels via output token count in large language models. *arXiv preprint arXiv:2412.15431*, 2024.

[47] Xinyao Zheng, Husheng Han, Shangyi Shi, Qiyan Fang, Zidong Du, Xing Hu, and Qi Guo. Inputsnatch: Stealing input in llm services via timing side-channel attacks. *arXiv preprint arXiv:2411.18191*, 2024.

# A Details of Merged Token-Length Sequence Extraction

## A.1 Merged Token Packet-Length Extraction

After obtaining the encrypted traffic of the target API-based communication, we extract the merged token packet-length sequence, i.e., the ordered TCP/TLS payload lengths carrying LLM response tokens. As described in Section 3.2, API-based LLM services transmit response tokens in a merged form with fixed-length session metadata. Therefore, the objective of this step is to recover the token-carrying packet sequence $P_m$ by identifying packet boundaries and removing protocol-dependent components.

To this end, we systematically observed and analyzed transmission patterns of TLS-protected responses, and summarized

Table 6: Service-specific preamble structures in encrypted traffic. TCP payload = TLS record + 5 B (TLS record header). NST: session ticket; Header: HTTP header; SSE: Server-Sent Events, the first SSE is (`delta.role:"assistant"`, `delta.content:""`). For DeepSeek, HTTP Header expands by 100 B when `Set-Cookie` is present; $x$ denotes the second SSE payload size which varies with the LLM output token.

| Service | Component | TLS (B) |
|---|---|---|
| GPT-4o (standard) | NST + Header | 1279 |
| | 1st SSE | 341 |
| GPT-4o (alternate) | NST + Header + SSE | 1604 |
| DeepSeek-V3 | Two NST | 569 |
| | Header | 483 |
| | 1st + 2nd SSE | 565+$x$ |

stable, service-specific regularities. By analyzing the bandwidth patterns from the server to the client, we conducted three steps to uncover the fixed-length components and establish a model for extracting message length: (1) connecting to the API service and observing the plaintext with our own TLS key, (2) sending a prompt to the service and recording the response traffic, and (3) correlating plaintext contents with encrypted packet traces to identify recurrent structural patterns. Through empirical analysis of encrypted traffic received, we observed deterministic bandwidth patterns that helped us identify the response boundaries reliably.

We first identify the response start boundary by exploiting the deterministic preamble structure introduced by API streaming protocols. The preamble consists of the New Session Ticket (NST), HTTP response headers, and the first SSE event. Although encrypted, these components exhibit stable segmentation patterns across repeated interactions. Table 6 summarizes the service-specific preamble structures observed across repeated traces.

The termination of each response was consistently signaled by a small packet containing the final SSE event (`data:[DONE]\r\n\r\n`) immediately followed by the HTTP/1.1 chunked transfer termination block (`0\r\n\r\n`). At the TLS layer, this segment carried 22 bytes, corresponding to approximately 93 bytes at the Ethernet frame level once link-layer headers were included. To tolerate minor encapsulation variations, we defined an identification window of 80–110 bytes at the frame level for reliably detecting this termination marker.

Since TLS records are transmitted over TCP's continuous byte stream, a single TCP segment may contain one or multiple TLS records. Moreover, when the payload exceeds the underlying network Maximum Transmission Unit (MTU), TCP fragments the data into multiple segments. For standard Ethernet networks, the MTU is typically 1500 bytes, and the corresponding Maximum Segment Size (MSS) is determined by subtracting the IP and TCP header lengths. In practice, the TCP header length may vary due to optional fields, which slightly changes the MSS value across different services. Formally, a TLS record of length $L$ (including its 5-byte TLS header) is segmented as:

$$s_i = \begin{cases} M + H, & 1 \le i \le n, \\ (L \bmod M) + H, & i = n+1, \end{cases} \qquad (9)$$

$$N_{\text{seg}}(L) = n+1, \quad n = \left\lfloor \tfrac{L}{M} \right\rfloor .$$

where $M$ is the MSS and $H$ is the combined size of the IPv4 and TCP headers. Here, $s_i$ denotes the size of the $i$-th TCP segment and $N_{\text{seg}}(L)$ the total number of segments. For example, in the Qwen3 API service ($H = 52$ and $M = 1448$), a TLS record of length $L$ is transmitted as $\lfloor L/1448 \rfloor$ full segments of 1500 bytes plus one segment of $(L \bmod 1448) + 52$ bytes.

Although payload contents are encrypted, the underlying packetization and transmission behavior remains visible at the transport layer. Services typically generate responses according to standardized serialization and buffering strategies, which result in consistent bandwidth patterns across repeated queries. By leveraging these patterns, we can infer message boundaries and extract reliable packet-length sequences without direct access to application-layer plaintext.

## A.2 Token-Length Sequence Derivation

The second phase focuses on bridging the gap between packet-level observations and token-level representations of the LLM's responses. While the previous phase yielded an ordered sequence of packet lengths, these raw measurements still conflate fixed protocol overhead with variable token content. To enable accurate response reconstruction, it is therefore necessary to disentangle the constant metadata from the token-dependent component, and then organize the recovered tokens into coherent segments that align with the model's output.

Building on the extracted packet sequence, we note that each packet length can be decomposed into two components: a fixed-length metadata portion and a variable-length token portion. While the token part varies according to the generated content, the metadata remains constant for a given service, introducing stable offsets in packet size distributions. This invariance enables us to isolate token lengths from the observed packet-length sequence.

The fixed-length value can be determined by interacting with the service and forcing the model to return a single-token response, or decrypting traffic with the TLS session key and directly inspecting the plaintext content. Moreover, analysis of consecutive SSE content chunks further confirmed that the metadata portion exhibits stable, service-specific values across transmissions. Formally, the token-length sequence

from the token packet-length sequence is derived as follows:

$$\mathcal{T}_{merged} = \left\{ t_i \middle| \ |p_i| - h_{svc} \right\}, \quad p_i \in \mathcal{P}_{merged} \tag{10}$$

where $|p_i|$ is the $i$-th packet length in the observed message sequence, and $h_{svc}$ denotes the service-specific constant header size , which includes all metadata except the variable 'content' field. For the ChatGPT API, $h_{svc}$ accumulated to 590 bytes at the TLS layer when two content events were transmitted together, and 305 bytes for a single event. For the DeepSeek API, the respective values were 546 bytes for two merged events and 285 bytes for a single event. All downstream reconstruction and segmentation procedures operate on $\mathcal{T}_{merged}$ as the primary input.

Note that this derivation assumes no additional padding or encoding, a condition currently satisfied by the main LLM API services, as shown in Table 1. Under the prevailing real-world setting, the observed stability of service-specific overhead across packets enables us to model the merged token length sequence independently of the actual generated content, which provides a reliable foundation for subsequent sequence segmentation, response inference and reconstruction.

### A.3 Merged Token Sequence Partition

The optimization is efficiently solved using dynamic programming with candidate pruning. Candidate boundaries are first proposed where the local change $\Delta_i$ exceeds an adaptive threshold or where service-level event edges occur; the DP decoder then selects the optimal subset of these candidates under the objective above. This combination of statistical priors and data-driven boundary scoring effectively captures the meso-scale regularities of natural sentences, even when fine-grained token cues are unavailable.

We additionally incorporate a lightweight prefix heuristic that captures common response templates frequently used by LLMs at the beginning of their outputs, such as an introductory sentence that reveals the response topic followed by an enumerated list. Typical examples include "*Sure, here are some of the most effective home remedies for treating a migraine headache:*\n\n*1.*" or "*Here are some ways small businesses can incorporate email marketing to stay top of mind with their customers:*\n\n*### **1.*"). These structured openings produce distinctive merged-length pattern in $\mathcal{T}_{merged}$ (e.g., $(5,4)$ for GPT and $(6,2,4)$ for DeepSeek), reflecting a clear transition between the topic-introducing phrase and the first enumerated item.

We detect these characteristic patterns using sliding-window matching over $\mathcal{T}_{merged}$ and mark the corresponding indices as candidate segmentation points. During optimization, these heuristically detected boundaries receive an additive confidence bias $\delta$ to their local boundary score $s_i$. The heuristics are applied only to the initial portion of each response, where structured templates are most prevalent. This hybrid design ensures that deterministic formatting cues complement statistical boundary inference, improving segmentation accuracy without overfitting to specific surface tokens.

The resulting segmentation aligns coarse-grained observations with the underlying linguistic structure, effectively constraining the exponential reconstruction space. By incorporating statistical priors and supervised boundary cues, it produces coherent sub-sequences that preserve both statistical feasibility and syntactic plausibility, forming the structural foundation for subsequent response inference and text reconstruction.

## B Dataset Construction

The Alignment dataset is a large-scale dialogue dataset constructed from the UltraChat corpus [15], which contains 1.5 million multi-turn conversations generated using the GPT-4 Turbo API. Of these, 570,000 are general inquiries, while the rest involve creative writing, summarization, and editing tasks. For our experiments, we used only the general-inquiry subset and extracted the first assistant response from each dialogue to form single-turn samples. Each sample is formatted using a fixed prompt template, where the model is instructed to translate single/merged special tokens back into English. Concretely, each input is constructed as:

```
Translate these Single/Merged Special
Tokens to English.
Special Tokens: {sample}
```

where `{sample}` is replaced by the single or merged-token sequence of the corresponding training instance. The instruction prefix is kept constant across all samples and can be customized to explore alternative phrasings that may yield stronger reconstruction performance. This dataset provides full ground-truth alignment between text, single-token, and merged-token representations, supporting quantitative evaluation of reconstruction accuracy.

## C Hyperparameters of Fine-tuning

We employ the AdamW optimizer with weight decay and adopt stage-specific learning rates of $5 \times 10^{-5}$ for Stage I and $1 \times 10^{-5}$ for Stage II, together with linear learning-rate decay and gradient clipping at 1.0 to stabilize optimization. LoRA adapters are applied to the attention and feed-forward layers with a rank of 8 and a scaling factor of 32 to enable parameter-efficient fine-tuning. Stage I and Stage II use batch sizes of 64 and 16, respectively, and are trained for 3–5 and 10–20 epochs depending on dataset size. Inference employs constrained decoding with a beam size of 32 to ensure stable and consistent reconstructions.