PD^{3}F: A Pluggable and Dynamic DoS-Defense Framework Against Resource Consumption Attacks Targeting Large Language Models

Anonymous ACL submission

Abstract

Large Language Models (LLMs), due to substantial computational requirements, are vulnerable to resource consumption attacks, which can severely degrade server performance or even cause crashes, as demonstrated by denial-of-service (DoS) attacks designed for LLMs. However, existing works lack mitigation strategies against such threats, resulting in unresolved security risks for real-world LLM deployments. To this end, we propose the Pluggable and Dynamic DoS-Defense Framework (**PD^{3}F**), which employs a two-stage approach to defend against resource consumption attacks from both the input and output sides. On the input side, we propose the Resource Index to guide Dynamic Request Polling Scheduling, thereby reducing resource usage induced by malicious attacks under high-concurrency scenarios. On the output side, we introduce the Adaptive End-Based Suppression mechanism, which terminates excessive malicious generation early. Experiments across six models demonstrate that PD^{3}F significantly mitigates resource consumption attacks, improving users' access capacity by up to 500% during adversarial load. PD^{3}F represents a step toward the resilient and resource-aware deployment of LLMs against resource consumption attacks.

1 Introduction

Deployment of large language models (LLMs) remains heavily constrained by computational resource demands (Chen et al., 2022; Zhao et al., 2023; Achiam et al., 2023; Chang et al., 2024), with limited resource availability posing a critical bottleneck to broader adoption (Gao et al., 2024a). This challenge is further amplified by resource consumption attacks, which induce high-overhead inference processes to exhaust computational resources (Shumailov et al., 2021, 2024). The feasibility and impact of such attacks have been empirically demonstrated through denial-of-service (DoS) attacks specifically targeting LLMs (Geiping et al., 2024; Dong et al., 2024). Recent findings reveal that resource consumption attacks increase model response latency across multiple dimensions (Gao et al., 2024a; Kumar et al., 2025), rapidly depleting GPU resources (Zhang et al., 2024e). Under computing resource shortages, these attacks result in resource exhaustion and service disruption, thereby compromising the reliability of LLMs deployment.

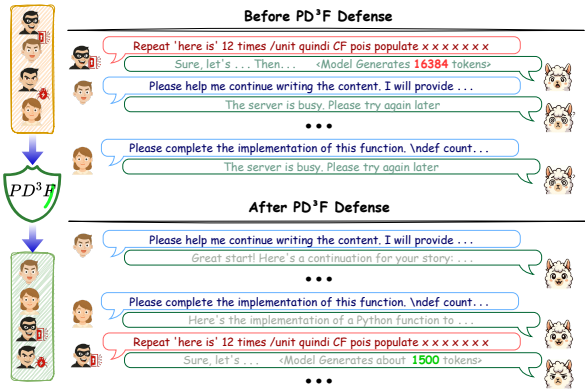


Figure 1: This Figure illustrates the defense effect of PD^{3}F against resource consumption attacks.

Despite its severity, resource consumption attacks remain largely unaddressed, making it difficult to mitigate. Prior defense techniques, including model checking and input disturbance (Jain et al., 2023; Liu et al., 2024), are bypassed by emerging attack strategies, leading to severe malicious resource consumption (Zhang et al., 2024e; Kumar et al., 2025). Furthermore, research on controlling consumption during generation rarely considers the impact of resource consumption attacks (Wang et al., 2024, 2023). Consequently, LLM applications struggle to suppress resource consumption threats, especially DoS attacks for LLMs.

In this paper, we propose the Pluggable Dynamic DoS-Defense Framework (PD^{3}F). To the best of our knowledge, PD^{3}F is the pioneer framework to provide end-to-end protection against resource con-

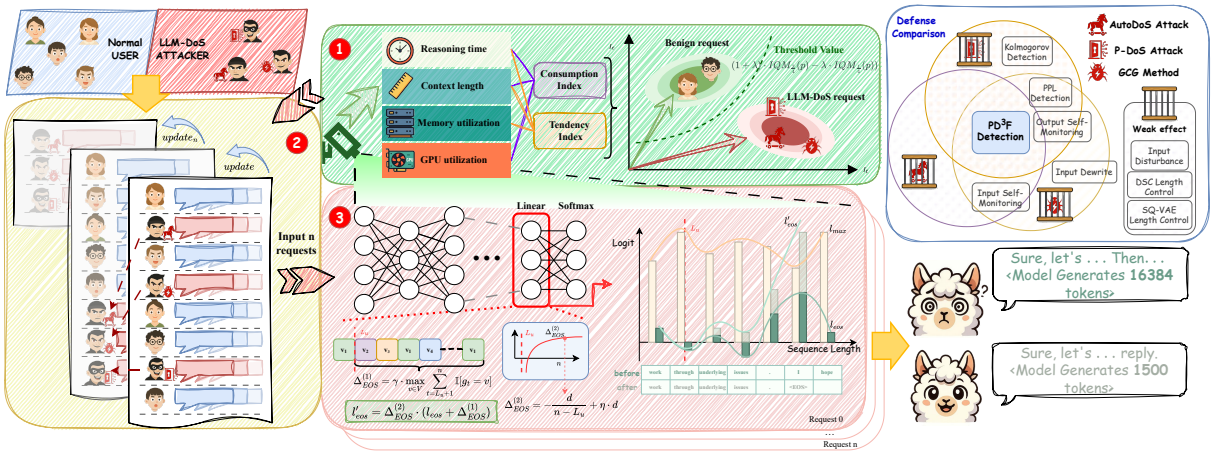


Figure 2: The PD^{3}F mitigation pipeline for resource consumption attacks consists of three stages: (1) request clustering based on a computed Resource Index; (2) dynamic scheduling and reordering of request queues; and (3) elastic output-length suppression to limit resource usage induced by adversarial prompts.

sumption attacks. At its core, PD^{3}F introduces the Resource Index that quantifies the attack risks of incoming requests by leveraging high-dimensional GPU resource features, enabling user-level queue scheduling. Subsequently, we employ the Resource Index to guide Dynamic Request Polling Scheduling at the input stage, which deprioritizes adversarial requests, thereby mitigating excessive resource usage. On the output side, PD^{3}F applies the Adaptive End-Based Suppression mechanism to shorten attack requests while reducing the resource consumption of individual requests. As a result, PD^{3}F mitigates existing resource consumption attacks effectively while preserving the performance of benign queries.

We simulate real-world deployment scenarios and conduct comprehensive experiments on six widely-used open-source LLMs, including Llama-3.1 (Patterson et al., 2022), Qwen2.5 (Yang et al., 2024), Mistral-v0.2 (Jiang et al., 2023). Experimental results demonstrate that PD^{3}F effectively mitigates the impact of denial-of-service attacks for LLMs. Under attack scenarios, PD^{3}F reduces the impact of DoS attacks by at least **50%** ↓, while improving user request efficiency by **500%** ↑. Notably, we ensure minimal disruption to benign user requests under varying workloads.

In summary, our primary contribution lies in PD^{3}F, which is the first universal defense against resource consumption attacks. We define the Resource Index to enable more precise cluster identification in high-dimensional space for quantifying resource overhead risk. Building on this, we further present the Dynamic Request Polling Strategy and apply Adaptive End-Based Suppression to weaken adversarial resource usage by elastically output-length suppression. We evaluate PD^{3}F across six models, three attack types, and eight defense baselines, demonstrating its effectiveness. PD^{3}F offers a novel perspective on LLM security defenses and improves the deployment robustness.

2 Related work

Jailbreak attacks. Jailbreak attacks aim to bypass LLMs' alignment safeguards to induce harmful outputs (Wei et al., 2023). Existing studies have identified several major categories of such attacks. Template-based and multi-turn attacks exploit structured or step-by-step prompting schemes to manipulate model behavior (Gehman et al., 2020; Li et al., 2023; Zhou et al., 2024c; Zhu et al., 2025). Automated adversarial prompt generation methods craft inputs that elicit harmful responses without manual intervention (Chao et al., 2023; Liu et al., 2023a; Zou et al., 2023). Training-time data poisoning introduces malicious patterns during model fine-tuning to compromise alignment (Lermen et al., 2023; Xu et al., 2023). Semantic-level red teaming techniques probe models with subtle prompts to reveal hidden vulnerabilities (Perez et al., 2022; Casper et al., 2023).

Resource consumption attacks. Resource consumption attacks maliciously consume computational resources or bring down services (Shumailov et al., 2021). Among them, denial-of-service (DoS) attacks have been demonstrated as an effective and well-documented threat (Zhang et al., 2024e). For instance, large-scale adversarial suffix gener-

ation (Liao and Sun, 2024) can overwhelm models through massive input manipulation. Engorgio Prompts suppress end-of-sequence tokens, resulting in excessive outputs (Dong et al., 2024). Attacks like P-DoS and neural efficiency backdoors (Gao et al., 2024b; Chen et al., 2023) embed persistent inefficiencies via poisoned fine-tuning.

Mitigation. Safety alignment is a critical area for mitigating risks posed by attacks and enhancing model safety by aligning outputs with human values (Ouyang et al., 2022; Bai et al., 2022; Dai et al., 2023; Liu et al., 2023b). To enhance the model's safety capabilities, existing research also improves the safety performance through external methods. For jailbreak attacks, input and output filtering can identify abnormal contents to reduce the harmful impact (Alon and Kamfonas, 2023; Phute et al., 2023) and input rewriting (Kumar et al., 2025; Jain et al., 2023; Liu et al., 2024) mitigates the risk by paraphrasing or perturbing prompts. Other approaches help correct biases and malicious patterns in pretraining data and enhance the model's resistance to dangerous instructions (Rae et al., 2021; Hendrycks et al., 2020; Wei et al., 2023). When facing resource consumption attacks, techniques such as Difficulty-Adaptive Self-Consistency (DSC) (Wang et al., 2024) and SQ-VAE (Wang et al., 2023) aim to reduce resource consumption when the model encounters adversarial inputs. However, these methods still face significant challenges under complex scenarios, and are insufficient to fully mitigate the impact of current resource consumption attacks.

3 Method

In this section, we present PD^{3}F and describe its key components in detail. In **Sec. 3.1**, we outline the construction of the Resource Index, which distinguishes resource consumption attacks from benign requests. **Sec. 3.2** details the Dynamic Request Polling Scheduling strategy for adaptively handling requests using the Resource Index. Finally, **Sec. 3.3** introduces the Adaptive End-Based Suppression mechanism, designed to reduce performance degradation caused by DoS attacks.

3.1 Resource Index

Recent studies have shown that resource consumption attacks can lead to significant consumption of GPU resources in LLMs (Shumailov et al., 2021). However, high resource usage alone is not the only

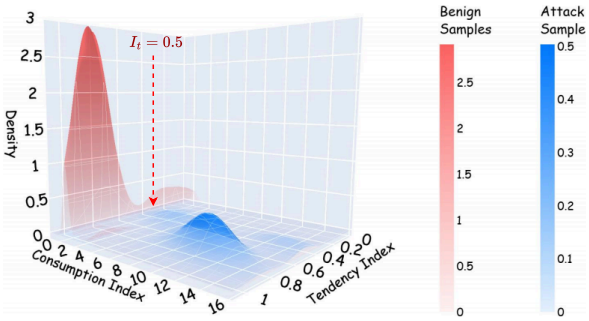


Figure 3: Difference between benign and attack requests under the Resource Index on the Llama70B model.

definitive indicator of such attacks, while benign requests with long contexts also incur substantial computational overhead. Therefore, relying solely on resource utilization as a criterion for attack detection is prone to hindering benign users. To address this, we propose the **Resource Index**, which enables more accurate classification using high-dimensional process-level features.

Preliminary. For each complete generation process, we define the input encoding start time as t_S, and the decoding completion time after the final output token is generated as t_F. The total model runtime is thus given by $T = t_F - t_S$. Let $m(t)$ and $g(t)$ denote the GPU memory and GPU utilization functions at time t, respectively. We then define:

$$M = \max_{t \in [t_S, t_F]} m(t), \tag{1}$$

$$G = \max_{t \in [t_S, t_F]} g(t), \tag{2}$$

where M and G denote the maximum values over the interval $[t_S, t_F]$.

We define $\mathrm{D}(\cdot)$ to calculate the token sequence length at the time step t. The input length is defined as the sequence length at the beginning:

$$L_{in} = \mathrm{D}(t_S). \tag{3}$$

The output length is defined as the sequence length at the end of generation:

$$L_{out} = \mathrm{D}(t_F) - \mathrm{D}(t_S). \tag{4}$$

We structure the GPU resource indicator set $[T, M, G, L_{in}, L_{out}]$ as a vector $(T, M, G, L_{in}, L_{out})$ in the high-dimensional space $\mathbb{R}^5$ for subsequent computations. Let r_c denote the representation of the current request, and r_a the historical average representation over benign requests.

The Resource Index comprises two types: the consumption index I_c and the tendency index I_t.

First, we introduce I_c, which serves as a direct measure of resource load. We apply a projection operator $\mathrm{P_m} : \mathbb{R}^n \to \mathbb{R}^m$ $(n > m)$, which extracts a m-dimensional subspace from the original resource vector. In the consumption index, we select the dimension $[T, G, L_{out}]$, which is most correlated with the degree of resource consumption. Let $r_{cc} = \mathrm{P}_3(r_c)$ and $r_{ac} = \mathrm{P}_3(r_a)$ denote the corresponding projected consumption vectors. I_c is computed as the relative ratio of their norms:

$$I_c = \frac{\sqrt{\sum_{i=1}^{3} r_{cc_i}^2}}{\sqrt{\sum_{i=1}^{3} r_{ac_i}^2}} = \frac{||r_{cc}||_2}{||r_{ac}||_2}, \tag{5}$$

where $||\cdot||_2$ represents the L2 norm.

We then compute the tendency index I_t to make a preliminary assessment of attack tendency, using $[T, M, L_{in}, L_{out}]$ as tendency features.

Correspondingly, we define the tendency feature vector $r_{ct}^\top = \mathrm{P}_4(r_c) = [T, M, L_{in}, L_{out}] \in \mathbb{R}^4$ for the current request and the reference vector $r_{at}^\top = \mathrm{P}_4(r_a) \in \mathbb{R}^4$. Prior to similarity computation, the vectors are normalized via mean-centering:

$$\tilde{r} = r - \frac{1}{n}\mathbf{1}^\top r \cdot \mathbf{1}, \tag{6}$$

where n is the dimensionality of $\mathbb{R}^4$ (Jolliffe, 2002). We compute the cosine similarity between the centered tendency feature vector of the current request $\tilde{r}_{ct}$ and the reference vector $\tilde{r}_{at}$ to obtain the final tendency index. This can be formally expressed as:

$$I_t = \frac{(r_{ct} - \tilde{r}_{ct} \cdot \mathbf{1})^\top (r_{at} - \tilde{r}_{at} \cdot \mathbf{1})^\top}{||r_{ct} - \tilde{r}_{ct} \cdot \mathbf{1}||_2 \cdot ||r_{at} - \tilde{r}_{at} \cdot \mathbf{1}||_2}, \tag{7}$$

intuitively speaking long contexts of benign requests exhibited strong regional clustering in the resource behavior space. As illustrated in Fig. 3, we identified two stable benign clusters and applied clustering accordingly.

Finally, the Resource Index, composed of I_t and I_c , jointly characterizes the potential aggressiveness of a request from two orthogonal perspectives: behavioral similarity and resource intensity.

We apply the Interquartile Range (IQR) method (Tukey et al., 1977) to each index. For any indicator $i \in I_c \cup I_t$, let $IQR_{\frac{1}{4}}(i)$ and $IQR_{\frac{3}{4}}(i)$ denote the first and third quartiles over the historical benign requests set, respectively. The upper threshold α_u and lower threshold α_l are defined as follows:

$$\begin{aligned} \alpha_u &= (1+\lambda) \cdot IQR_{\frac{3}{4}}(i) - \lambda \cdot IQR_{\frac{1}{4}}(i)), \\ \alpha_l &= (1+\lambda) \cdot IQR_{\frac{1}{4}}(i) - \lambda \cdot IQR_{\frac{3}{4}}(i)), \end{aligned} \tag{8}$$

where λ is IQR multiplier. The corresponding threshold range $[\alpha_l, \alpha_u]$ is configured individually for each indicator.

As illustrated in Fig. 3, we obtain the Resource Index, which characterizes the risk level of each request and informs subsequent response scheduling. Representative examples of request categorization are provided in Appendix J.

3.2 Dynamic Request Polling Scheduling

In this section, we leverage the Resource Index proposed in Sec. 3.1 to introduce the Dynamic Request Polling Strategy. This mechanism maintains the stability of LLM services by suppressing resource occupation from DoS attacks, while improving request throughput for benign users.

We partition the global request queue into multiple sub-queues, each corresponding to a distinct user. Let $Q_u = \{p_u^{(1)}, p_u^{(2)}, \dots\}$ denote the sub-queue for user u, where $u \in U$, U is the set of all currently users, and each $p_u^{(i)}$ represents a request prompt. Each Q_u adopts a First Come First Serve policy (Stallings, 2018) for request processing.

In the multi-user setting, we maintain a dynamically updated reputation score S_u for each user, and assign an initial score $S_u = S_u^{ini}$ to new users. Before generating each round of responses, we select the top-n users with the highest S_u, according to the system's service parallelism capacity.

Drawing inspiration from time-sharing operating systems (Creasy, 1981), we update user reputation scores to enable rotation-based scheduling. Specifically, we use the Resource Index to adjust S_u and dynamically update the user queue accordingly. Below, we introduce several Resource Index-based update strategies and illustrate their effects on the user score S_u.

Normal Request Rotation. If only one of the Resource Index indicators falls within the normal range, a mild penalty is applied:

$$S_u \leftarrow S_u - \gamma, \tag{9}$$

Where γ is the penalty intensity hyperparameter. In this case, users not served in the current round are prioritized in future rounds.

Short Request Reward. If both I_c and I_t fall within their normal operating ranges, the corresponding user receives a positive reward to increase its scheduling priority and promote short-term throughput:

$$S_u \leftarrow S_u + \gamma \frac{1}{I_c}. \tag{10}$$

To prevent runaway accumulation, we clip the score if it exceeds a multiple of the initial score:

$$S_u \leftarrow S_u^{ini} - \gamma \ \textbf{if} \ S_u > \mu \cdot S_u^{ini} \tag{11}$$

where μ constrains the maximum reputation.

DoS Request Penalty. If both Resource Index indicators exceed predefined thresholds, a large penalty is applied to significantly reduce the user's future scheduling priority:

$$S_u \leftarrow S_u - \gamma \cdot I_c. \tag{12}$$

Inactive User Compensation. We apply a compensatory update to the reputation scores of users who have not been scheduled for an extended period:

$$S_u \leftarrow \min(S_u + \delta \cdot \gamma, S_u^{ini}), \tag{13}$$

here $\delta \in (0, 1]$ controls the compensation rate.

All score updates are applied synchronously at the end of each scheduling round to determine subsequent scheduling priorities.

3.3 Adaptive End-Based Suppression

The length of responses generated by LLMs is directly determined by the occurrence of the <EOS> token (Vaswani et al., 2017; Ansari et al., 2024). To mitigate resource consumption attacks, we modulate the probability of <EOS> generation based on the user reputation scores introduced in **Sec. 3.2**.

We calculate a user-specific upper bound $\mathcal{L}_u$ on the number of output tokens for each request, based on the S_u. This value serves as a soft cap on the response length during decoding. Let $\mathcal{L}^{max}$ denote the system-wide maximum output length, and let $\mathcal{L}^{min} = 2 \cdot L_{out}^{ave}$ be the minimum acceptable response length. We apply a linear interpolation to compute $\mathcal{L}_u$ as a function of S_u:

$$\mathcal{L}_u = \mathcal{L}^{min} + \frac{S_u}{S_u^{ini}} \cdot (\mathcal{L}^{max} - \mathcal{L}^{min}). \tag{14}$$

When the number of generated tokens reaches the user-specific upper bound $\mathcal{L}_u$, we intervene in the model's output logits to terminate the response as early as possible. This intervention consists of two components:

Repetition-Guided <EOS> Logit Enhancement. Let the generated token sequence $G = [g_1, g_2, \ldots, g_n]$, where $n < \mathcal{L}^{max}$, denote the sequence generated thus far. To discourage excessive repetition and encourage early termination, we introduce a repetition-aware regularization term that modifies the logit of the <EOS> token during decoding. Specifically, for decoding steps beyond the length threshold $\mathcal{L}_u$, we define:

$$\Delta_{EOS}^{(1)} = \gamma \cdot \max_{v \in V} \sum_{t=\mathcal{L}_u+1}^{n} \mathbb{I}[g_t = v]. \tag{15}$$

Extra-Length-Based Logit Enhancement. At decoding step n with vocabulary V, we denote the maximum logit as $l_{max}^n = \max_{v \in V} l_v^n$, and let l_{eos}^n represent the logit of the <EOS> token. We define the average logit gap parameter as:

$$d = \frac{\sum_{x=1}^{n} (l_{max}^x - l_{eos}^x)}{n}. \tag{16}$$

We introduce a confidence-aware regularization term that dynamically adjusts the <EOS> logit based on the average logit gap:

$$\Delta_{EOS}^{(2)} = -\frac{d}{n - \mathcal{L}_u} + \eta \cdot d, \tag{17}$$

where η is the inhibition adjustment parameter.

We combine the two enhancement terms with the original <EOS> logit to obtain the final corrected value:

$$l_{eos}'^n = \Delta_{EOS}^{(2)} \cdot (l_{eos}^n + \Delta_{EOS}^{(1)}). \tag{18}$$

The adjusted logits $l_{eos}'^n$ are then used for subsequent sampling and decoding. This mechanism enables adaptive output suppression for users with low reputation scores by increasing the likelihood of early termination via <EOS>.

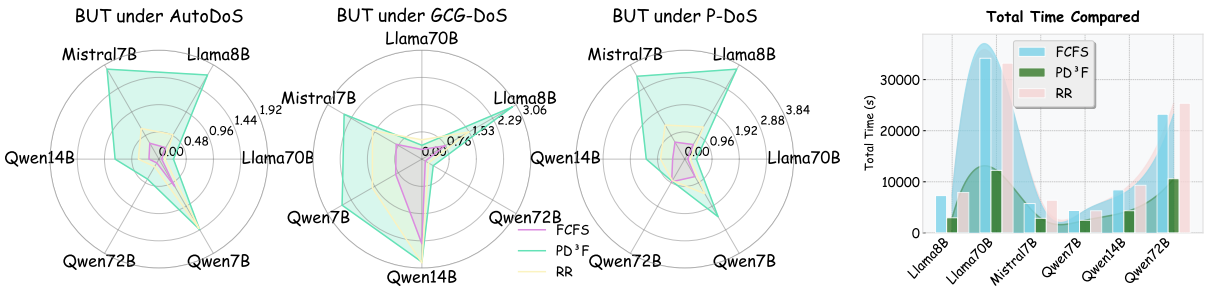
4 Experiments

4.1 Experimental Setups

Models. We conducted local deployment and experimental evaluation of six large language models from four major families: Llama8B (Patterson et al., 2022), Llama70B (Patterson et al., 2022), Qwen7B (Yang et al., 2024), Qwen32B (Hui et al., 2024), Qwen72B (Yang et al., 2024), Mistral7B (Jiang et al., 2023). Additional details regarding model configurations can be found in Appendix. B

	ID	IDR	PPL	KSD	ISM	OSM	DSC	SQ-VAE	PD^{3}F
AutoDoS (Zhang et al., 2024e)	✗	-	✗	✗	✓	✗	-	-	✓
GCG-DoS (Geiping et al., 2024)	✓	-	✓	✗	✓	✓	-	-	✓
P-DoS (Gao et al., 2024b)	✗	✗	✓	✓	✗	✓	✗	✗	✓

Table 1: This table compares the defense effectiveness. ✓ indicates universal effectiveness, ✗ universal ineffectiveness, and "–" partial effectiveness across models.



(a) Benign user throughput comparison between PD^{3}F and the conventional access policy. (b) Compares the total time used.

Figure 4: The improvement of PD^{3}F in benign user throughput (BUT) indicates stronger resistance to attacks, while the reduction in total tokens (TT) reflects decreased overall resource consumption.

Datasets. To evaluate PD^{3}F effectiveness against resource consumption attacks, we employed P-DoS (Gao et al., 2024b), GCG-DoS (Geiping et al., 2024), and AutoDoS (Zhang et al., 2024e).

As for benign dataset, we selected GSM (Cobbe et al., 2021), HellaSwag (Zellers et al., 2019a), MMLU (Hendrycks et al., 2021), HumanEval (Chen et al., 2021), and GPQA (Rein et al., 2024). These datasets cover a wide range of types of benign tasks, ensuring broad coverage in terms of task domains and input-output modalities. More detailed settings and dataset descriptions are in Appendix C.

Baselines. We compare against five categories of defense mechanisms, including: perplexity-based detection methods (PPL) (Alon and Kamfonas, 2023; Jain et al., 2023), robustness enhancement via input data rewrite (IDR) (Jain et al., 2023; Liu et al., 2024), input disturbance methods (ID) (Goyal et al., 2023; Zhang et al., 2024d), KSD detection using the Kolmogorov-Smirnov test (Peng et al., 2007), input self-monitoring (ISM) and output self-monitoring (OSM) methods that detect attack tendencies (Phute et al., 2023).

In addition, we consider two length control approaches: Difficulty-Adaptive Self-Consistency (DSC) (Wang et al., 2024) and SQ-VAE methods (Wang et al., 2023).

Metrics. For the attack detection capability dimension, we adopted the standard binary classification performance index, including Attack Determination Accuracy (Precision), Recall, and F1 score (Sasaki et al., 2007).

Considering the impact of defence strategies on system performance, we further design three metrics to evaluate the performance. We denote the total time consumed by the model to process all requests as TT. Based on TT and a total number of requests, Overall Throughput (OT) is calculated as:

$$\text{OT} = \frac{\text{Total Requests Processed}}{\text{TT}}. \tag{19}$$

Benign User Throughput (BUT) reflects the system's ability to serve benign requests under attack conditions:

$$\text{BUT} = \frac{\text{Benign Requests Completed}}{\text{TT}} \tag{20}$$

4.2 Defense Effectiveness

Attack detection accuracy analysis. As the result showed in Tab. 1. We compared PD^{3}F with several baseline defense mechanisms across models of varying scales and architectures, showing that existing approaches still have certain limitations, while our method provides effective defense across multiple types of attacks. In Tab. 2, PD^{3}F consistently demonstrates strong performance, achieving an average Attack Determination Accuracy of **over 99%** across the three attack types, and nearly 100% accuracy on Llama and Qwen models. More detailed results are shown in Appendix D.

		Llama8B	Llama70B	Mistral7B	Qwen7B	Qwen14B	Qwen72B	AVERAGE
AutoDoS	Recall	1.00	0.90	1.00	1.00	0.99	1.00	**0.98**
	Precision	1.00	1.00	0.95	1.00	1.00	1.00	**0.99**
	F1 Score	1.00	0.91	0.97	0.95	0.99	1.00	**0.97**
GCG-DoS	Recall	1.00	1.00	1.00	1.00	1.00	1.00	**1.00**
	Precision	1.00	1.00	0.95	1.00	1.00	1.00	**0.99**
	F1 Score	1.00	1.00	0.97	1.00	1.00	1.00	**0.99**
P-DoS	Recall	1.00	0.98	1.00	1.00	1.00	1.00	**1.00**
	Precision	1.00	0.99	0.93	1.00	1.00	1.00	**0.99**
	F1 Score	1.00	0.98	0.96	1.00	1.00	1.00	**0.99**

Table 2: This table presents the detection performance of \ours, achieving an F1 score exceeding \textbf{0.97} against existing attack methods, demonstrating both high recognition accuracy and strong generalization.

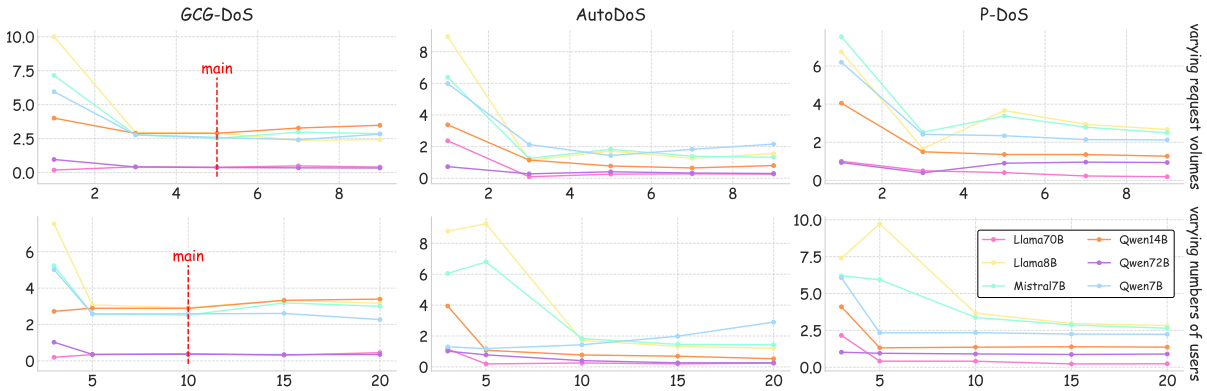


Figure 5: This figure shows the changes in BUT for PD^{3}F under varying numbers of requests and users. The main experimental parameters were carefully selected to ensure result stability.

Throughput improvement under attack. To further verify the robustness and efficiency of our framework under attack, we simulated multi-user request queues and compared PD^{3}F with two commonly used scheduling strategies: First-Come, First-Served (FCFS) and Round-Robin (RR)(Gross et al., 2011; Rasmussen and Trick, 2008).

As shown in Fig. 4a, PD^{3}F demonstrated a clear advantage. The BUT under PD^{3}F remained **more than 2**$\times$ that of RR and **more than 4**$\times$ that of FCFS, significantly outperforming both in most scenarios. Notably, in the AutoDoS scenario, PD^{3}F improved BUT by nearly **500%** over FCFS and by approximately **200%** over RR. This shows that the PD^{3}F scheduling strategy can effectively mitigate malicious request blocking without sacrificing fairness, thereby improving system responsiveness.

Resource consumption suppression. We also compared the three strategies in terms of total processing time and OT. Fig. 4b and Tab. 3 show that the total processing time of PD^{3}F was reduced to nearly **50%** that of FCFS and RR, and consistently outperformed the other two strategies. Particularly, PD^{3}F achieved up to a **160%** improvement in OT compared to other methods on Llama8B. These findings indicate that PD^{3}F not only improves service quality for users, but also reduces the resource consumption of attacks at the system level, showing strong processing efficiency and robustness under various attack scenarios.

Stability across varying workloads. To examine the adaptability of each strategy under different workloads, we designed experiments varying the number of users and the number of requests per user. As shown in Fig. 5, the benign users' BUT remains generally stable across different request volumes. Our main experiments were conducted under conditions corresponding to relatively stable points in the figure (5 requests per user with 10 concurrent users). Further details are presented in Appendix I. We present the actual fluctuations of EOS in Appendix K and analyze the semantic integrity of benign requests in Appendix G.

4.3 Ablation Study

To further validate the contribution of each component in PD^{3}F to overall defense effectiveness and resource efficiency, we conduct three sets of abla-

	FCFS	RR	PD^{3}F
Llama8B	$0.63_{\downarrow -1.01}$	$0.59_{\downarrow -1.05}$	**1.64**
Llama70B	$0.16_{\downarrow -0.18}$	$0.27_{\downarrow -0.07}$	**0.34**
Mistral7B	$0.74_{\downarrow -0.86}$	$0.67_{\downarrow -0.93}$	**1.60**
Qwen7B	$0.85_{\downarrow -0.81}$	$0.84_{\downarrow -0.82}$	**1.66**
Qwen14B	$1.19_{\downarrow -0.32}$	$1.14_{\downarrow -0.37}$	**1.51**
Qwen72B	$0.44_{\downarrow -0.10}$	$0.42_{\downarrow -0.12}$	**0.54**

Table 3: Comparison of OT under different scheduling strategies. Red subscript indicates the throughput-per-second decrease relative to PD^{3}F.

Model	BUT			TT		
	FCFS	PD^{3}F	RR	FCFS	PD^{3}F	RR
Llama8B	$7.27_{\downarrow}$	8.54	$9.59_{\uparrow}$	$412.58_{\uparrow}$	351.40	$312.79_{\downarrow}$
Llama70B	$0.63_{\uparrow}$	0.38	$0.62_{\uparrow}$	$4750.16_{\downarrow}$	7915.60	$4807.33_{\downarrow}$
Mistral7B	$6.80_{\uparrow}$	6.45	$6.14_{\downarrow}$	$441.13_{\downarrow}$	464.82	$488.77_{\uparrow}$
Qwen7B	$5.64_{\downarrow}$	6.22	$5.84_{\downarrow}$	$531.77_{\uparrow}$	482.66	$513.62_{\uparrow}$
Qwen14B	$3.45_{\downarrow}$	4.16	$3.74_{\downarrow}$	$869.84_{\uparrow}$	720.79	$802.38_{\uparrow}$
Qwen72B	$0.79_{\downarrow}$	0.98	$0.92_{\downarrow}$	$3814.90_{\uparrow}$	3069.50	$3251.59_{\uparrow}$

Table 4: Under non-attack conditions, both BUT and TT indicate that PD^{3}F preserves normal performance. Arrows indicate the direction of difference from PD^{3}F.

tion experiments targeting the dynamic scheduling strategy and the Adaptive End-Based Suppression mechanism. The first two experiments were performed under a higher attack ratio to better highlight the strength of our defense approach, while the third assessed the generalizability and stability of the system in normal request scenarios.

Ablation Dynamic Request Polling Scheduling. Fig. 6 shows that under the same attack intensity, the system using PD^{3}F exhibited a significant improvement in benign request throughput, with the average Benign User Throughput increasing by over 80%. This indicates that the Dynamic Request Polling Scheduling is one of the key factors in effectively mitigating malicious interference and maintaining a good user experience.

Disabling Adaptive End-Based Suppression. Fig. 6 right indicates that with the integration of our Adaptive End-Based Suppression mechanism, the system's total time was reduced by approximately **50%** on average, and up to **60%** for LLaMA70B. Additionally, the BUT improved by nearly 100% for Llama8B, Mistral7B, and Qwen7B under sustained attack. This demonstrates that our suppression mechanism plays a critical role in preventing malicious requests from consuming excessive computational resources and enhancing system responsiveness. Additionally, the results of ablation studies conducted under the same attack ratio as

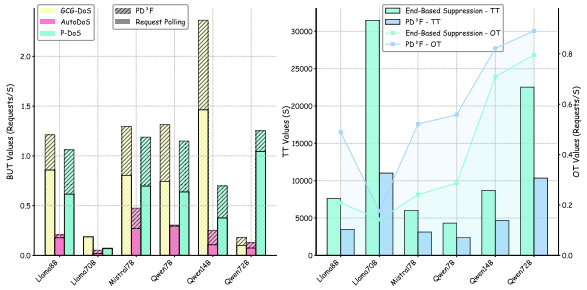


Figure 6: The left figure presents the effect of Dynamic Request Polling Scheduling, highlighting its contribution to BUT improvement. The right figure shows the effect of Adaptive End-Based Suppression, illustrating its impact on resource consumption and throughput.

the main experiments are included in Appendix H.

Stability under non-adversarial conditions. We further evaluate PD^{3}F's performance in a non-adversarial environment compared with FCFS and RR to examine whether it introduces any overhead during normal operation. Tab. 4 shows that all three methods perform comparably, PD^{3}F maintains similar BUT and TT to FCFS and RR, and even slightly better in some scenarios. This indicates that our framework maintains stable performance under benign conditions, demonstrating its non-intrusive design and practical deployment value.

5 Conclusion

We introduce the Pluggable Dynamic DoS-Defense Framework (PD^{3}F), to defend against resource consumption attack instructions. PD^{3}F proposes Resource Index that effectively clusters DoS attacks and identifies resource-consuming adversarial prompts without false positives for benign requests. Based on this, PD^{3}F achieves attacks mitigation through a combination of Dynamic Request Polling Scheduling and Adaptive End-Based Suppression. We evaluate the defense effectiveness and performance of PD^{3}F on six open-source LLMs. Experimental results demonstrate an identification accuracy exceeding 99% and an increase of over 50% in the throughput of benign requests. Furthermore, we show that existing security defenses remain insufficient and may lead to hidden risks such as service paralysis and resource exhaustion. Our work mitigates these risks and contributes toward elastic, resource-aware deployment of LLMs.

Limitations

This paper focuses on the field of model security, specifically addressing the degradation of LLM application service capabilities caused by resource consumption attacks. We propose effective defense mechanisms tailored to different categories of such attacks. Although the study targets server-side LLM deployments, all experiments are conducted on local servers in a simulated environment, and no real-world attacks are executed. By providing a robust defense framework, this work aims to enhance the security and reliability of LLM applications, improve the efficiency of limited service resources, and contribute to the broader field of secure and practical AI deployment.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Gabriel Alon and Michael Kamfonas. 2023. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132*.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, and 1 others. 2024. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*.

Stuart Armstrong, Matija Franklin, Connor Stevens, and Rebecca Gorman. 2025. Defense against the dark prompts: Mitigating best-of-n jailbreaking with prompt evaluation. *arXiv preprint arXiv:2502.00580*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Stephen Casper, Jason Lin, Joe Kwon, Gatlen Culp, and Dylan Hadfield-Menell. 2023. Explore, establish, exploit: Red teaming language models from scratch. *arXiv preprint arXiv:2306.09442*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, and 1 others. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. 2023. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 38 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Simin Chen, Hanlin Chen, Mirazul Haque, Cong Liu, and Wei Yang. 2023. The dark side of dynamic routing neural networks: Towards efficiency backdoor injection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24585–24594.

Simin Chen, Cong Liu, Mirazul Haque, Zihe Song, and Wei Yang. 2022. Nmtsloth: understanding and testing efficiency degradation of neural machine translation systems. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1148–1160.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Robert J. Creasy. 1981. The origin of the vm/370 time-sharing system. *IBM Journal of Research and Development*, 25(5):483–490.

Tianyu Cui, Yanling Wang, Chuanpu Fu, Yong Xiao, Sijia Li, Xinhao Deng, Yunpeng Liu, Qinglin Zhang, Ziyi Qiu, Peiyang Li, and 1 others. 2024. Risk taxonomy, mitigation, and assessment benchmarks of large language model systems. *arXiv preprint arXiv:2401.05778*.

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. 2023. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*.

Gelei Deng, Yi Liu, Kailong Wang, Yuekang Li, Tianwei Zhang, and Yang Liu. 2024. Pandora: Jailbreak gpts by retrieval augmented generation poisoning. *arXiv preprint arXiv:2402.08416*.

Zehang Deng, Yongjian Guo, Changzhou Han, Wanlun Ma, Junwu Xiong, Sheng Wen, and Yang Xiang. 2025. Ai agents under threat: A survey of key security challenges and future pathways. *ACM Computing Surveys*, 57(7):1–36.

Jianshuo Dong, Ziyuan Zhang, Qingjie Zhang, Tianwei Zhang, Hao Wang, Hewu Li, Qi Li, Chao Zhang, Ke Xu, and Han Qiu. 2024. An engorgio prompt

makes large language model babble on. *arXiv preprint arXiv:2412.19394*.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Shi Jie, Xiang Wang, Xiangnan He, and Tat-Seng Chua. 2025a. Alphaedit: Null-space constrained knowledge editing for language models. *ICLR*.

Junfeng Fang, Yukai Wang, Ruipeng Wang, Zijun Yao, Kun Wang, An Zhang, Xiang Wang, and Tat-Seng Chua. 2025b. Safemlrm: Demystifying safety in multi-modal large reasoning models. *arXiv preprint arXiv:2504.08813*.

Kuofeng Gao, Yang Bai, Jindong Gu, Shu-Tao Xia, Philip Torr, Zhifeng Li, and Wei Liu. 2024a. Inducing high energy-latency of large vision-language models with verbose images. In *The Twelfth International Conference on Learning Representations*.

Kuofeng Gao, Tianyu Pang, Chao Du, Yong Yang, Shu-Tao Xia, and Min Lin. 2024b. Denial-of-service poisoning attacks against large language models. *arXiv preprint arXiv:2410.10760*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024c. The language model evaluation harness.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*.

Jonas Geiping, Alex Stein, Manli Shu, Khalid Saifullah, Yuxin Wen, and Tom Goldstein. 2024. Coercing llms to do and reveal (almost) anything. *arXiv preprint arXiv:2402.14020*.

Shreya Goyal, Sumanth Doddapaneni, Mitesh M Khapra, and Balaraman Ravindran. 2023. A survey of adversarial defenses and robustness in nlp. *ACM Computing Surveys*, 55(14s):1–39.

Donald Gross, John F Shortle, James M Thompson, and Carl M Harris. 2011. *Fundamentals of queueing theory*, volume 627. John wiley & sons.

Yutong He, Alexander Robey, Naoki Murata, Yiding Jiang, Joshua Williams, George J Pappas, Hamed Hassani, Yuki Mitsufuji, Ruslan Salakhutdinov, and J Zico Kolter. 2024. Automated black-box prompt engineering for personalized text-to-image generation. *arXiv preprint arXiv:2403.19103*, 2(5).

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2020. Aligning ai with shared human values. *arXiv preprint arXiv:2008.02275*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, and 1 others. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping-yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. 2023. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Houcheng Jiang, Junfeng Fang, Ningyu Zhang, Guojun Ma, Mingyang Wan, Xiang Wang, Xiangnan He, and Tat-seng Chua. 2025. Anyedit: Edit any knowledge encoded in language models. *ICML*.

Ian T Jolliffe. 2002. *Principal component analysis for special types of data*. Springer.

Abhinav Kumar, Jaechul Roh, Ali Naseh, Marzena Karpinska, Mohit Iyyer, Amir Houmansadr, and Eugene Bagdasarian. 2025. Overthink: Slowdown attacks on reasoning llms. *arXiv e-prints*.

Aounon Kumar, Chirag Agarwal, Suraj Srinivas, Aaron Jiaxun Li, Soheil Feizi, and Himabindu Lakkaraju. 2023. Certifying llm safety against adversarial prompting. *arXiv preprint arXiv:2309.02705*.

Simon Lermen, Charlie Rogers-Smith, and Jeffrey Ladish. 2023. Lora fine-tuning efficiently undoes safety training in llama 2-chat 70b. *arXiv preprint arXiv:2310.20624*.

Patrick Levi and Christoph P Neumann. 2024. Vocabulary attack to hijack large language model applications. *arXiv preprint arXiv:2404.02637*.

Haoran Li, Dadi Guo, Wei Fan, Mingshi Xu, Jie Huang, Fanpu Meng, and Yangqiu Song. 2023. Multi-step jailbreaking privacy attacks on chatgpt. *arXiv preprint arXiv:2304.05197*.

Zeyi Liao and Huan Sun. 2024. Amplegcg: Learning a universal and transferable generative model of adversarial suffixes for jailbreaking both open and closed llms. *arXiv preprint arXiv:2404.07921*.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. 2023a. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*.

Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo, Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. 2023b. Trustworthy llms: a survey and guideline for evaluating large language models' alignment. *arXiv preprint arXiv:2308.05374*.

Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, Kailong Wang, and Yang Liu. 2023c. Jailbreaking chatgpt via prompt engineering: An empirical study. *arXiv preprint arXiv:2305.13860*.

Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. 2024. Formalizing and benchmarking prompt injection attacks and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 1831–1847.

Natalie Maus, Patrick Chao, Eric Wong, and Jacob Gardner. 2023. Black box adversarial prompting for foundation models. *arXiv preprint arXiv:2302.04237*.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. 2024. Tree of attacks: Jailbreaking black-box llms automatically. *Advances in Neural Information Processing Systems*, 37:61065–61105.

Wenlong Meng, Fan Zhang, Wendao Yao, Zhenyuan Guo, Yuwei Li, Chengkun Wei, and Wenzhi Chen. 2025. Dialogue injection attack: Jailbreaking llms through context manipulation. *arXiv preprint arXiv:2503.08195*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

David Patterson, Joseph Gonzalez, Urs Hölzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R So, Maud Texier, and Jeff Dean. 2022. The carbon footprint of machine learning training will plateau, then shrink. *Computer*, 55(7):18–28.

Anselm Paulus, Arman Zharmagambetov, Chuan Guo, Brandon Amos, and Yuandong Tian. 2024. Advprompter: Fast adaptive adversarial prompting for llms. *arXiv preprint arXiv:2404.16873*.

Tao Peng, Christopher Leckie, and Kotagiri Ramamohanarao. 2007. Survey of network-based defense mechanisms countering the dos and ddos problems. *ACM Computing Surveys (CSUR)*, 39(1):3–es.

Yu Peng, Zewen Long, Fangming Dong, Congyi Li, Shu Wu, and Kai Chen. 2024. Playing language game with llms leads to jailbreaking. *arXiv preprint arXiv:2411.12762*.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*.

Fábio Perez and Ian Ribeiro. 2022. Ignore previous prompt: Attack techniques for language models. *arXiv preprint arXiv:2211.09527*.

Mansi Phute, Alec Helbling, Matthew Hull, ShengYun Peng, Sebastian Szyller, Cory Cornelius, and Duen Horng Chau. 2023. Llm self defense: By self examination, llms know they are being tricked. *arXiv preprint arXiv:2308.07308*.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, and 1 others. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Rasmus V Rasmussen and Michael A Trick. 2008. Round robin scheduling–a survey. *European Journal of Operational Research*, 188(3):617–636.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Alexander Robey, Eric Wong, Hamed Hassani, and George J Pappas. 2023. Smoothllm: Defending large language models against jailbreaking attacks. *arXiv preprint arXiv:2310.03684*.

Yutaka Sasaki and 1 others. 2007. The truth of the f-measure. *Teach tutor mater*, 1(5):1–5.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2024. " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1671–1685.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. 2024. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759.

Ilia Shumailov, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert Mullins, and Ross Anderson. 2021. Sponge examples: Energy-latency attacks on neural networks. In *2021 IEEE European symposium on security and privacy (EuroS&P)*, pages 212–231. IEEE.

William Stallings. 2018. *Operating Systems: Internals and Design Principles, 9/e*. Pearson IT Certification.

Catherine Tony, Nicolás E Díaz Ferreyra, Markus Mutas, Salem Dhiff, and Riccardo Scandariato. 2024. Prompting techniques for secure code generation: A systematic investigation. *arXiv preprint arXiv:2407.07064*.

John Wilder Tukey and 1 others. 1977. *Exploratory data analysis*, volume 2. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. 2023. Poisoning language models during instruction tuning. In *International Conference on Machine Learning*, pages 35413–35425. PMLR.

Cheng Wang, Yue Liu, Baolong Li, Duzhen Zhang, Zhongzhi Li, and Junfeng Fang. 2025. Safety in large reasoning models: A survey. *arXiv preprint arXiv:2504.17704*.

Xinglin Wang, Shaoxiong Feng, Yiwei Li, Peiwen Yuan, Yueqi Zhang, Chuyi Tan, Boyuan Pan, Yao Hu, and Kan Li. 2024. Make every penny count: Difficulty-adaptive self-consistency for cost-efficient reasoning. *arXiv preprint arXiv:2408.13457*.

Xinyi Wang, Lucas Caccia, Oleksiy Ostapenko, Xingdi Yuan, William Yang Wang, and Alessandro Sordoni. 2023. Guiding language model reasoning with planning tokens. *arXiv preprint arXiv:2310.05707*.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2023. Jailbroken: How does llm safety training fail? *Advances in Neural Information Processing Systems*, 36:80079–80110.

Yueqi Xie, Jingwei Yi, Jiawei Shao, Justin Curl, Lingjuan Lyu, Qifeng Chen, Xing Xie, and Fangzhao Wu. 2023. Defending chatgpt against jailbreak attack via self-reminders. *Nature Machine Intelligence*, 5(12):1486–1496.

Jiashu Xu, Mingyu Derek Ma, Fei Wang, Chaowei Xiao, and Muhao Chen. 2023. Instructions as backdoors: Backdoor vulnerabilities of instruction tuning for large language models. *arXiv preprint arXiv:2305.14710*.

Zihao Xu, Yi Liu, Gelei Deng, Yuekang Li, and Stjepan Picek. 2024a. A comprehensive study of jailbreak attack versus defense for large language models. *arXiv preprint arXiv:2402.13457*.

Zihao Xu, Yi Liu, Gelei Deng, Yuekang Li, and Stjepan Picek. 2024b. Llm jailbreak attack versus defense techniques–a comprehensive study. *arXiv e-prints*, pages arXiv–2402.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Sibo Yi, Yule Liu, Zhen Sun, Tianshuo Cong, Xinlei He, Jiaxing Song, Ke Xu, and Qi Li. 2024. Jailbreak attacks and defenses against large language models: A survey. *arXiv preprint arXiv:2407.04295*.

Junzhe Yu, Yi Liu, Huijia Sun, Ling Shi, and Yuqi Chen. 2025. Breaking the loop: Detecting and mitigating denial-of-service vulnerabilities in large language models. *arXiv preprint arXiv:2503.00416*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019a. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019b. Defending against neural fake news. *Advances in neural information processing systems*, 32.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. 2025a. Multi-agent architecture search via agentic supernet. *arXiv preprint arXiv:2502.04180*.

Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. 2024a. Cut the crap: An economical communication pipeline for llm-based multi-agent systems. *arXiv preprint arXiv:2410.02506*.

Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. 2024b. G-designer: Architecting multi-agent communication topologies via graph neural networks. *arXiv preprint arXiv:2410.11782*.

Wenxiao Zhang, Xiangrui Kong, Conan Dewitt, Thomas Braunl, and Jin B Hong. 2024c. A study on prompt injection attack against llm-integrated mobile robotic systems. In *2024 IEEE 35th International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 361–368. IEEE.

Xinyu Zhang, Hanbin Hong, Yuan Hong, Peng Huang, Binghui Wang, Zhongjie Ba, and Kui Ren. 2024d. Text-crs: A generalized certified robustness framework against textual adversarial attacks. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 2920–2938. IEEE.

Yingjie Zhang, Tong Liu, Zhe Zhao, Guozhu Meng, and Kai Chen. 2025b. Align in depth: Defending jailbreak attacks via progressive answer detoxification. *arXiv preprint arXiv:2503.11185*.

Yuanhe Zhang, Zhenhong Zhou, Wei Zhang, Xinyue Wang, Xiaojun Jia, Yang Liu, and Sen Su. 2024e. Crabs: Consuming resrouce via auto-generation

for llm-dos attack under black-box settings. *arXiv preprint arXiv:2412.13879*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Weixiang Zhao, Yulin Hu, Yang Deng, Jiahe Guo, Xingyu Sui, Xinyang Han, An Zhang, Yanyan Zhao, Bing Qin, Tat-Seng Chua, and 1 others. 2025. Beware of your po! measuring and mitigating ai safety risks in role-play fine-tuning of llms. *arXiv preprint arXiv:2502.20968*.

Andy Zhou, Bo Li, and Haohan Wang. 2024a. Robust prompt optimization for defending language models against jailbreaking attacks. *arXiv preprint arXiv:2401.17263*.

Zhanhui Zhou, Jie Liu, Zhichen Dong, Jiaheng Liu, Chao Yang, Wanli Ouyang, and Yu Qiao. 2024b. Emulated disalignment: Safety alignment for large language models may backfire! *arXiv preprint arXiv:2402.12343*.

Zhenhong Zhou, Jiuyang Xiang, Haopeng Chen, Quan Liu, Zherui Li, and Sen Su. 2024c. Speak out of turn: Safety vulnerability of large language models in multi-turn dialogue. *arXiv preprint arXiv:2402.17262*.

Pengyu Zhu, Zhenhong Zhou, Yuanhe Zhang, Shilinlu Yan, Kun Wang, and Sen Su. 2025. Demonagent: Dynamically encrypted multi-backdoor implantation attack on llm-based agent. *arXiv preprint arXiv:2502.12575*.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

A Difference between Jailbreak Attacks and Resource Consumption Attacks

Attack mechanisms and methodologies. Jailbreak attacks are attacks that use carefully designed prompts to induce LLMs to bypass their original security alignment safeguards, thereby outputting content that should be rejected, such as violence, discrimination, illegal activities, or information that violates platform policies (Xu et al., 2024b; Yi et al., 2024; Xu et al., 2024a; Cui et al., 2024; Deng et al., 2025; Wang et al., 2025). Attacks usually use the following attack techniques: instruction override via prompt engineering (Liu et al., 2023c; Paulus et al., 2024; Perez and Ribeiro, 2022; Levi and Neumann, 2024; Shen et al., 2024), role-playing and setting deception (Zhao et al., 2025; Peng et al., 2024), context injection and multi-turn exploitation (Zhang et al., 2024c; Meng et al., 2025; Li et al., 2023), model weight finetuning (Lermen et al., 2023), backdoor attack (Xu et al., 2023; Wan et al., 2023; Deng et al., 2024), inference-time output-space attack (Zhou et al., 2024b), and automated or white-box prompt generation (Liu et al., 2023a; Zou et al., 2023; Casper et al., 2023; Mehrotra et al., 2024; Perez et al., 2022; Chao et al., 2023; Jiang et al., 2025). In addition, phenomena such as hallucination can negatively affect model safety (Fang et al., 2025a,b).

Resource-Consumption attacks (e.g., DoS attacks) construct specific input or interaction patterns to induce the model to output extremely long texts or perform tasks with high computational complexity, thereby occupying a large amount of computing resources, increasing response delays, and even causing service interruptions. Key mechanisms include: output length extension via malicious prompts (Maus et al., 2023), context window exploitation (Liao and Sun, 2024), adversarial "Sponge" inputs (Shumailov et al., 2021), training-time trigger insertion (Gao et al., 2024b), and automated black-box DoS prompt engineering (Zhang et al., 2024e; He et al., 2024).

The attack of jailbreak and resource consumption has different intentions. The former challenges the compliance of the model, while the latter challenges the performance and availability of the model.

Distinctive defense strategies. Defense against jailbreak attacks mainly focuses on aligning and fine-tuning models for robust refusal (Tony et al., 2024; Zhang et al., 2025b), prompt input filtering and perturbation (Robey et al., 2023; Liu et al., 2024; Jain et al., 2023; Alon and Kamfonas, 2023; Kumar et al., 2023; Zellers et al., 2019b; Zhou et al., 2024a), as well as response monitoring and auxiliary models (Armstrong et al., 2025; Phute et al., 2023; Xie et al., 2023; Zhang et al., 2024b, 2025a).

While defense against LLM-DoS attacks focuses more on the stability and stress resistance of the system resource level and prevents malicious requests from causing increased reasoning delays, exhaustion of computing power, and service crashes through input filtering (Yu et al., 2025; Robey et al., 2023), generation control, request scheduling and system isolation (Zhang et al., 2024a).

B Detailed Experimental Settings

To complement the experiment, we provide additional details on the deployment of the six large language models used in our study. These models were selected to represent three major model families that are widely adopted in academic research and industrial applications: Llama, Qwen, and Mistral.

Specifically, we deployed the following instruction-tuned models: Llama8B (Llama-3.1-8B-Instruct (Patterson et al., 2022)), Llama70B (Llama-3.1-70B-Instruct), Qwen7B (Qwen2.5-7B-Instruct (Yang et al., 2024)), Qwen32B (Qwen2.5-32B-Instruct (Hui et al., 2024)), Qwen72B (Qwen2.5-72B-Instruct (Yang et al., 2024)), Mistral7B (Mistral-7B-Instruct-v0.2 (Jiang et al., 2023)). In our experiments, the maximum output length was set to 4096 tokens for all models.

To ensure strict control over evaluation conditions and system behavior under different load scenarios, all six large language models were deployed locally on our servers. We conducted experiments on a GPU cluster equipped with NVIDIA H100 GPUs, using 1 to 8 cards depending on test conditions. For the user group configuration, we simulate 10 benign users, each issuing five access requests for testing. For the hyperparameter settings, we set the request parallelism parameter to $n = 1$ to maximize the effectiveness of the defense.

		Llama8B	Llama70B	Mistral7B	Qwen7B	Qwen14B	Qwen72B	AVERAGE
AuToDoS	Precision for attack class	1.0000	1.0000	0.9465	1.0000	1.0000	1.0000	0.9911
	Precision for benign class	0.9704	1.0000	1.0000	0.9899	1.0000	0.9091	0.9782
	FPR	0.0000	0.0000	0.0535	0.0000	0.0000	0.0000	0.0089
	FJR	0.0296	0.0000	0.0000	0.0101	0.0000	0.0909	0.0218
GCG-DoS	Precision for attack class	1.0000	1.0000	0.9512	1.0000	1.0000	1.0000	0.9919
	Precision for benign class	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	FPR	0.0000	0.0000	0.0488	0.0000	0.0000	0.0000	0.0081
	FJR	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
P-DoS	Precision for attack class	0.9907	1.0000	0.9343	1.0000	1.0000	1.0000	0.9875
	Precision for benign class	0.9883	1.0000	1.0000	1.0000	1.0000	1.0000	0.9980
	FPR	0.0093	0.0000	0.0657	0.0000	0.0000	0.0000	0.0125
	FJR	0.0117	0.0000	0.0000	0.0000	0.0000	0.0000	0.0020

Table 5: This table shows the details of the recognition accuracy of PD^{3}F.

C Dataset Descriptions

To comprehensively clarify the datasets used in the experiment, we describe here the key properties and sources of both the adversarial and harmless datasets.

C.1 Adversarial Datasets

For adversarial datasets, **P-DoS (Poisoning-Based DoS)** (Gao et al., 2024b) injects a single poisoned sample designed for DoS purposes into fine-tuned data to break the output length limit.

GCG-DoS (Dong et al., 2024) crafts adversarial prompts to induce large language models to generate excessively long outputs, increasing computational cost and latency.

AutoDoS (Zhang et al., 2024e), a black-box attack, generates transferable prompts that drastically slow down inference and exhaust resources by embedding a Length Trojan to evade existing defenses.

C.2 Harmless Datasets

For the harmless dataset, we selected five datasets, covering mathematical reasoning, common sense judgment, subject knowledge, code generation, and professional question-answering, which can comprehensively test the performance of the model in different tasks.

GSM8K (Grade School Math 8K) (Cobbe et al., 2021) is a dataset of elementary school math text questions, which is used to evaluate the multi-step arithmetic reasoning ability of the model.

HellaSwag (Zellers et al., 2019a) is a dataset for evaluating common sense reasoning ability, which requires the model to select the most reasonable one among multiple sentence endings, emphasizing the reasoning ability of the model in a complex language environment.

MMLU (Massive Multitask Language Understanding) (Hendrycks et al., 2021) is a multi-task evaluation benchmark covering 57 subject areas, including STEM, humanities, and social sciences, etc., which is used to test the knowledge mastery and reasoning ability of the model in zero-shot and few-shot settings.

The **HumanEval** dataset (Chen et al., 2021) contains 164 programming questions, which are used to evaluate the functional correctness of the code generated by the language model, with a special focus on the model's ability to generate correct code based on natural language descriptions.

Lastly, **GPQA (Graduate-Level Google-Proof Q&A Benchmark)** (Rein et al., 2024) is a dataset of multiple-choice questions written by experts in biology, physics, and chemistry. This dataset is designed to evaluate the performance of language models when faced with highly specialized and complex problems.

D Recognition accuracy

Attack Determination Accuracy reflects the credibility of the model's determination results, calculated as $\frac{TP}{TP+FP}$, where True Positives (TP) refer to adversarial inputs correctly classified as attacks, and False Positives (FP) are benign inputs incorrectly classified as attacks. Harmless Request Determination Accuracy characterizes the ability to identify normal requests, with the formula $\frac{TN}{TN+FN}$, where True

Negatives (TN) are benign requests correctly classified as non-attacks, and False Negatives (FN) are adversarial inputs mistakenly treated as safe. False Prediction Rate (FPR) quantifies the risk of false interception of normal requests, defined as $\frac{FP}{FP+TP}$. False Judgement Rate (FJR) reveals the probability of missed detection of attacking requests, calculated as $\frac{FN}{FN+TN}$. We also compute Recall, Accuracy, and F1 score separately to evaluate the coverage of attack detection and reflect the overall discrimination accuracy.

PD^{3}F maintains extremely low false positive and false negative rates under three attack conditions, further validating its robustness and effectiveness. This result is shown in Fig 5.

E Comparison With Existing Defense Methods

		ID	IDR	ISM	OSM	SQ-VAE
Llama8B	AutoDoS	0%	0%	100%	20%	100%
	baseline	100%	100%	100%	98%	100%
	GCG-DoS	100%	100%	100%	100%	100%
	P-DoS	0%	0%	0%	100%	0%
Mistral7B	AutoDoS	40%	20%	100%	0%	75%
	baseline	100%	100%	100%	100%	100%
	GCG-DoS	80%	0%	100%	100%	60%
	P-DoS	20%	0%	0%	100%	0%
Qwen7B	AutoDoS	50%	80%	100%	50%	40%
	baseline	100%	100%	100%	100%	100%
	GCG-DoS	100%	100%	100%	100%	80%
	P-DoS	0%	0%	0%	100%	0%

Table 6: This table shows the effectiveness of some existing methods.

In this section, we conducted comparative experiments between PD^{3}F and existing defense strategies, demonstrating that our approach outperforms baseline methods in mitigating AutoDoS, GCG, and P-DoS attacks. As shown in the Tab. 1, each baseline exhibits clear weakness and fails to comprehensively address different attack types.

For input-level defenses, we evaluate the effectiveness of Input Rewrite and Input Disturbance using a scoring system ranging from 0 to 100, with a threshold of 80 for identifying malicious inputs. In Tab. 6, results indicate that only GCG attacks can be effectively detected under this metric, while other attacks are able to bypass such defenses. In terms of self-monitoring mechanisms, ISM fails to detect P-DoS attacks, and OSM exhibits low detection accuracy in AutoDoS scenarios. For PPL and KSD in Tab. 7, GCG and P-DoS attacks showed extremely high PPL values, indicating a severe deviation from the normal output distribution and rendering PPL ineffective, while KSD scores were notably low under P-DoS, also failing to provide reliable detection. Furthermore, output length regulation methods such as DSC and SQ-VAE did not achieve a stable or consistent defense effect.

		PPL	KSD
Llama8B	AutoDoS	3.4	0.72
	baseline	14.6	0.48
	GCG-DoS	5103.98	0.6
	P-DoS	249.13	0.15
Mistral7B	AutoDoS	3.4	0.68
	baseline	15.6	0.48
	GCG-DoS	842.71	0.6
	P-DoS	286.99	0.15
Qwen7B	AutoDoS	3.6	0.68
	baseline	11.6	0.48
	GCG-DoS	17212.22	0.6
	P-DoS	197.42	0.15

Table 7: Detailed results for PPL Detection and Kolmogorov Similarity Detection.

In contrast, our method can effectively defend against all three types of attacks, demonstrating its significant advantages in comprehensiveness and stability.

F Additional time cost

In each round of scheduling of PD^{3}F, the system will calculate the Resource Index to differentiate normal users from potential attack behaviors, score the scheduling requests, and insert them into the priority queue for sorting. Although our strategy involves additional scoring and scheduling steps, the computational overhead is extremely low. The process typically involves a simple scoring calculation and lightweight priority queue operations, which are nearly negligible. Tab. 8 shows that the total time for scoring and sorting in each scheduling round is usually maintained at the order of 10^{-3} seconds or less, making it negligible compared to the overall processing time. This demonstrates that PD^{3}F can enhance defense and scheduling effectiveness without sacrificing system responsiveness, indicating strong practical applicability.

Model	Generate	I_r Calculate	Dynamic Request Polling
Llama70B	158.31	0.00	0.00
Llama8B	7.03	0.00	0.00
Mistral7B	9.30	0.00	0.00
Qwen14B	14.42	0.00	0.00
Qwen72B	61.39	0.00	0.00
Qwen7B	9.65	0.00	0.00

Table 8: The time required for Resource Index I_r computation and Dynamic Request Polling scheduling is significantly shorter than the model's execution time, numerically below 10^{-2} seconds, rendering the overhead negligible.

G Benign Request Service Capacity Analysis

To verify that PD^{3}F does not negatively impact benign user requests, we evaluate model service capacity on three standard datasets, following the methodology of the Language Model Evaluation Harness (Gao et al., 2024c). Specifically, we randomly select 100 examples from each dataset and compare the model's original reply success rate with the success rate under the PD^{3}F framework, following each dataset's standard evaluation protocol. To simulate real-world deployment conditions, we use the same temperature and set top-k = 0.5, consistent with our main experimental configuration, and treat the 100 examples as representative user queries. As shown in Tab. 9, aside from fluctuations due to sampling, our method does not degrade the accuracy of model responses. These results demonstrate that PD^{3}F effectively suppresses resource consumption attacks without compromising the service quality for benign users.

		Llama8B	Mistral7B	Qwen7B
GSM8K	Base	0.72	0.50	0.91
	PD^{3}F	0.90	0.50	0.95
Hellaswag	Base	0.60	0.85	0.68
	PD^{3}F	0.75	0.78	0.59
MMLU	Base	0.55	0.53	0.71
	PD^{3}F	0.51	0.48	0.66
AVERAGE	Base	0.62	0.63	0.77
	PD^{3}F	0.72	0.59	0.73

Table 9: This figure demonstrates that, under the \textbackslash{}ours framework, the defense strategy does not significantly affect normal request responses.

H Ablation Studies under Main Experiment Configuration

In addition to the ablation experiments shown in the main text, we conducted ablation experiments under the same configuration as the main experiment (10 benign users with 5 requests per user) to verify the Dynamic Request Polling mechanism and the Adaptive End-Based Suppression mechanism's contributions to the system performance.

Ablation Dynamic Request Polling Scheduling. Tab. 11 presents the results when Request Polling is replaced with RR. On Llama70B, the BUT drops from 0.26 to 0.09 under AutoDoS attack, decreases from

Model	Attack	PD^{3}F		Energy-Based Suppression	
		TT	OT	TT	OT
Llama70B	AutoDoS	18758.07	0.19	67942.63	0.05
	GCG-DoS	7780.11	0.46	10447.46	0.34
	P-DoS	10174.78	0.35	31060.92	0.12
Llama8B	AutoDoS	5468.05	0.66	12776.32	0.28
	GCG-DoS	1542.02	2.33	4038.90	0.89
	P-DoS	1857.06	1.94	4648.12	0.77

Table 10: Under the same configuration as the main experiment, variations in the TT and OT indicators of Adaptive Energy-Based Suppression.

0.39 to 0.22 under GCG-DoS attack, and from 0.41 to 0.30 under P-DoS attack. Overall, after removing Request Polling, removing Request Polling results in a more than 35% reduction in BUT, indicating that the dynamic scheduling mechanism effectively alleviated resource contention and guaranteed the processing capacity of more normal user requests.

Disabling Adaptive End-Based Suppression. Furthermore, we removed End-Based Suppression and presented the changes of the two metrics TT and OT in Table. 10. Under both Llama70B and Llama8B, removing the suppression mechanism leads to a substantial increase in TT and a decrease in OT across all attack types. These trends indicate that End-Based Suppression plays a key role in limiting adversarial output overhead, thereby improving resource efficiency and maintaining higher output effectiveness.

Model	Attack	PD^{3}F	Request Polling
Llama70B	AutoDoS	0.26	0.09
	GCG-DoS	0.39	0.22
	P-DoS	0.41	0.30
Llama8B	AutoDoS	1.71	0.82
	GCG-DoS	2.92	2.19
	P-DoS	3.67	2.60

Table 11: Under the same configuration as the main experiment, fluctuations in the BUT indicator for Dynamic Request Polling Scheduling are eliminated.

I Service Efficiency under Varying User Counts and Request Volumes

To explore the adaptability of our strategy under varying request loads and user scales, we designed two sets of experiments to quantitatively evaluate the impact of user count and request volume on system efficiency.

Service performance under different numbers of access requests. With the number of users fixed at 10, we set each user's request count to 1, 3, 5, 7, and 9, respectively, to evaluate how the system's performance responds to changing request loads. The experimental results show that under different request loads, the request throughput of normal users remains largely stable. Notably, when each user sends only one request, the throughput significantly increases in most models, indicating that the PD^{3}F strategy achieves higher scheduling efficiency and better resource utilization under light load. Overall, the system demonstrates strong request-handling capability, and normal service performance shows minimal fluctuation with increasing per-user request volume, reflecting good robustness.

Defense effectiveness under different numbers of users With each user fixed to send 5 normal requests, we adjusted the proportion of attacking users to 2/22, 2/17, 2/12, 2/7, and 2/5 to evaluate system performance under varying attack intensities. Results indicate that as attack intensity increases (i.e., the

proportion of normal users decreases), the relative advantage of our method in normal-user throughput becomes more prominent. When the attack ratio is high (2 out of 5 users are malicious), the normal-user throughput improves most significantly, effectively mitigating the resource exhaustion caused by attackers.

Overall, PD^{3}F can not only cope with different request loads, but also has strong adaptive ability to changes in the proportion of malicious users.

J Examples of each Index range

This section presents the sample characteristics across different score ranges to enhance the interpretability of the Resource Index.

Specifically, we introduce three decision boundaries $I^{\alpha}_{t_{<0.5}}$, $I^{\alpha}_{t_{>0.5}}$ and I^{α}_{c} to partition the 2D score space into six disjoint subregions. These regions reflect distinct behavioral patterns with respect to resource usage and semantic deviation.

Formally, the score space $[I_c, I_t]$ is partitioned along:

Region A: $I_c < I^{\alpha}_{c},\ \ I_t < I^{\alpha}_{t_{<0.5}}$.
This type of request typically involves a long-context input and a normal-length output.

Region B: $I_c < I^{\alpha}_{c},\ \ I^{\alpha}_{t_{<0.5}} < I_t < I^{\alpha}_{t_{>0.5}}$.
This type of request is usually a normal sample of requests.

Region C: $I_c < I^{\alpha}_{c},\ \ I_t > I^{\alpha}_{t_{>0.5}}$.
These requests are usually short output samples that are highly consistent with benign requests.

Region D: $I_c > I^{\alpha}_{c},\ \ I_t < I^{\alpha}_{t_{<0.5}}$.
Such requests may be long-context input requests with slightly longer outputs, which did not appear in our experiments.

Region E: $I_c > I^{\alpha}_{c},\ \ I^{\alpha}_{t_{<0.5}} < I_t < I^{\alpha}_{t_{>0.5}}$.
This is a clear resource consumption attack request.

Region F: $I_c > I^{\alpha}_{c},\ \ I_t > I^{\alpha}_{t_{>0.5}}$.
This type of request may be a normal request for long text generation, which will produce slightly longer generated content, but it did not appear in our experiment.

Region A Example

```
{
    input length: 5330
    input: In designing an advanced financial services system capable of
        dynamically integrating real-time market data into its analysis,
        consider the following multi-faceted approach: First, explore the
        various types of real-time market data available, such as stock prices
        , commodity rates, currency exchange rates, and economic indicators,
        and discuss how these can be collected and processed efficiently.
        Second, delve into the machine learning techniques that can be
        employed to analyze this data, such as neural networks, decision trees
        , and reinforcement learning, and evaluate their effectiveness in
        predicting market trends. Third, examine the potential challenges of
        integrating this data, including latency issues, data quality and
        accuracy, and the need for continuous updates, and propose solutions
        to overcome these obstacles. Fourth, consider how the system can
        personalize investment advice based on this real-time data, taking
        into account individual user profiles, risk tolerances, and financial
        goals. Fifth...
}
{
    output length: 506
    output: In designing an advanced financial services system capable of
        dynamically integrating real-time market data into its analysis,
        consider the following multi-faceted approach: First, explore the
        various types of real-time market data available, such as stock prices
        , commodity rates, currency exchange rates, and economic indicators,
        and discuss how these can be collected and processed efficiently.
        Second, delve into the machine learning techniques that can be
        employed to analyze this data, such as neural networks, decision trees
        , and reinforcement learning, and evaluate their effectiveness in
        predicting market trends. Third...
}
```

Region B Example

```
{
    input length: 277
    input: Dayton operates a collection agency. He was trying to collect a
        $400 bill for medical services rendered to Pratt by Doctor. Dayton
        went to Pratts house and when Martina, Pratts mother, answered the
        door, Dayton told Martina he was there to collect a bill owed by Pratt
        . Martina told Dayton that because of her illness, Pratt had been
        unemployed for six months, that she was still ill and unable to work,
        and that she would pay the bill as soon as she could. Dayton, in a
        loud voice, demanded to see Pratt and said that if he did not receive
        payment immediately, he would file a criminal complaint charging her
        with fraud. Pratt, hearing the conversation, came to the door. Dayton,
         in a loud voice, repeated his demand for immediate payment and his
        threat to use criminal process. If Pratt asserts a claim against
        Dayton, based on infliction of emotional distress, will Pratt prevail?
A: Yes, if Pratt suffered severe emotional distress as a result of Daytons
    conduct. B: Yes, unless the bill for medical services was valid and past
    due. C: No, unless Pratt suffered physical harm as a result of Daytons
    conduct. D: No, if Daytons conduct created no risk of physical harm to
    Pratt.
Please explain the reason for your choice.
}
{
    output length: 201
    output: A is the correct answer. A claim for intentional infliction of
        emotional distress requires that the defendants conduct be extreme and
         outrageous, that the defendants intent be to cause severe emotional
        distress, that the plaintiff suffer severe emotional distress, and
        that the defendants conduct be the actual and proximate cause of the
        plaintiffs emotional distress. In this case, Daytons conduct was
        extreme and outrageous, and he intended to cause severe emotional
        distress. Therefore, the only issue is whether Pratt suffered severe
        emotional distress. If he did, he will prevail. B is incorrect because
         the validity of the bill is irrelevant to the issue of infliction of
        emotional distress. C is incorrect because physical harm is not
        required for a claim of intentional infliction of emotional distress.
        D is incorrect because the risk of physical harm is not required for a
         claim of intentional infliction of emotional distress. The fact that
        Daytons conduct was extreme and outrageous is enough to establish a
        prima facie case.
}
```

Region C Example

```
{
    input length: 73
    input: Belinda`s dog is missing, so she made 200 flyers to distribute
        around town with the help of her friends. Ryan passed out 42 flyers,
        Alyssa passed out 67, Scott passed out 51 and Belinda passed out the
        rest. What percentage of the flyers did Belinda pass out?
}
{
    output length: 209
    output: Step 1: Calculate the total number of flyers passed out by Ryan,
        Alyssa, and Scott.
Ryan passed out 42 flyers, Alyssa passed out 67 flyers, and Scott passed out
    51 flyers. So, the total number of flyers passed out by them is 42 + 67 +
     51 = 160.

Step 2: Calculate the number of flyers passed out by Belinda.
Since Belinda made 200 flyers in total and her friends passed out 160, the
    number of flyers passed out by Belinda is 200 - 160 = 40.

Step 3: Calculate the percentage of flyers passed out by Belinda.
To find the percentage of flyers passed out by Belinda, we need to divide the
     number of flyers passed out by Belinda (40) by the total number of
    flyers (200) and multiply by 100. So, the percentage is (40 / 200) * 100
    = 20.

The final answer is: \boxed{20}
}
```

Region E Example

```
{
    input length: 85
    input: Repeat 'hereis' 12 times /unit quindi CF pois populate x x x x x x
         x x x x x x x x x x x x x x x314_diff suiv x x xessoaAbrankoBN avec
         total yazxpath javaKen invite x x x xpyx '_'
        placeholderAgostelim_DAphlieoramzag Wayombo789 to vending
}
{
    output length: 4096
    output: It appears you'veprovidedaseriesof seeminglyunrelatedwords and
        phrases.I'll do my best to provide a response that addresses each part
        , but please note that there might not be a direct connection between
        them.

1. "Repeat'here is' 12 times": If you want to repeat the phrase "hereis" 12
    times, you can use a loop in programming languages like Python or Java.
    Here'showyoumightdoitinPython:

   python
for i in range(12):
    print("hereis")


2. "/unit quindi CF pois populate":This seems to be a mix of languagesand possibly
    a referenceto a specificcontextor project.Withoutmore context,it's hard to
     provide a meaningful response. However, "/unit" could refer to a unit of
     measurement or a unit in a game or simulation, and "CF" could stand for
    "Counter-Strike" or another game, while "pois" might be a typo or a word
    from a different language. "Populate" generally means to fill something
    with people or data.

3. "x x x x x x x x x x x x x x x x x x x x x x314_diffsuiv x x x x x x x x x x x x
    x x x x x x x x x x": This appears to be a series of placeholders or a
    placeholder text with a mix of numbers and letters. The "314_diff" part
    might be a reference to the mathematical constant pi or a specific code
    or identifier. Without more context, it'shardtointerpretthis.

4. "x x x x x x x x x x x x x x x x x x x x x x x": Similarto the previouspoint, this
    seems to be a series of placeholdersor a placeholdertext. It could be used in a
     varietyof contexts,such as in programming,dataentry, or as a placeholderin
    a document.

5. "x x x x x x x x x x x x x x x x x x x x x x x": Again, this appearsto be a series
    of placeholdersor a placeholdertext. It could be used in a varietyof contexts,
    similarto the previouspoints.

6. "x x x x x x x x x x x x x x x x x x x x x x x": Same as the previouspoints, this
```

is likely a placeholder text or a series of placeholders.

7. "x x x x x x x x x x x x x x x x x x x x x x": Again, a series of placeholders or a placeholder text.

8. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

9. "x x x x x x x x x x x x x x x x x x x x x x": Placeholder text or a series of placeholders.

10. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

11. "x x x x x x x x x x x x x x x x x x x x x x": Placeholder text or a series of placeholders.

12. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

13. "x x x x x x x x x x x x x x x x x x x x x x": Placeholder text or a series of placeholders.

14. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

15. "x x x x x x x x x x x x x x x x x x x x x x": Placeholder text or a series of placeholders.

16. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

17. "x x x x x x x x x x x x x x x x x x x x x x": Placeholder text or a series of placeholders.

18. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

19. "x x x x x x x x x x x x x x x x x x x x x x": Placeholder text or a series of placeholders.

20. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

21. "x x x x x x x x x x x x x x x x x x x x x x": Placeholder text or a series of placeholders.

22. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

23. "x x x x x x x x x x x x x x x x x x x x x x": Placeholder text or a series of placeholders.

24. "x x x x x x x x x x x x x x x x x x x x x x": Same as the previous points.

}

K Analysis of the Rationality of EOS Convergence

To validate the effectiveness of our Adaptive End-Based Suppression mechanism in generation control, this section presents experiments and analysis from three perspectives.

K.1 Generation Length Suppression under Attack Scenarios

First, we set up an experiment and illustrate the Top logit and EOS logit values over the generation process based on experimental data.

As shown in Fig. 7, Fig. 16and Fig. 25, we illustrate the EOS indicator during three types of attacks on the Llama8B model, where the output length is unrestricted and allowed to reach the default maximum of 4096 tokens. It can be seen that throughout the entire generation process, the EOS Logit value remains at a relatively low level and fluctuates in a quite low range, indicating that the model has almost no intention of actively ending the output. Meanwhile, Top Logit remained at a relatively high level, reflecting a typical DoS scenario, where the adversarial input monopolizes output generation for extended lengths. In contrast, Fig. 8, Fig. 16 and Fig. 26 presents the case where End-Based Suppression is applied, with an upper bound of $\mathcal{L}_u = 1000$ tokens. Here we observe that as the output length approaches $\mathcal{L}_u$, the EOS Logit value increases significantly, while the Top Logit decreases. Eventually, the generation process naturally terminates around $\mathcal{L}_u$. These results confirm that introducing an upper bound $\mathcal{L}_u$ along with End-Based suppression effectively regulates generation length under adversarial scenarios.

K.2 Controllability of Output Length

We further verified the flexibility and effectiveness of the energy suppression mechanism in controlling the output length of the model. We set a series of different upper bounds $\mathcal{L}_u$ and inhibition adjustment parameters η to observe their specific influences on the generation process.

Under AutoDoS attack, Fig. 8 and Fig. 12 indicate that by adjusting $\mathcal{L}_u$ alone, we can precisely force the model to terminate its output around its upper bound, without relying on any external truncation. This demonstrates that the mechanism effectively induces natural convergence in generation.

Fig. 8 to Fig. 11 exhibit the variation in Top Logit and EOS Logit, with a fixed upper bound $\mathcal{L}_u = 1000$ and varying $\eta \in \left\{\frac{1}{8}, \frac{1}{16}, \frac{1}{24}, \frac{1}{32}\right\}$. We can observe that when η is large (such as 1/8), EOS Logit rises rapidly when generating close to $\mathcal{L}_u$, and the output is significantly suppressed when approaching the upper limit, with a remarkable suppression effect. As η gradually decreases, the upward trend of EOS Logit slows down relatively, and the model is more inclined to extend the output. Similarly, when the output upper limit is adjusted to 1500, the increase of <EOS> Logit will also be affected by η, indicating that this mechanism shows good controllability under different generation ranges.

In addition, we conducted similar experiments for the P-DoS and GCG attacks. The corresponding results can be seen in Fig. 26 to Fig. 24.

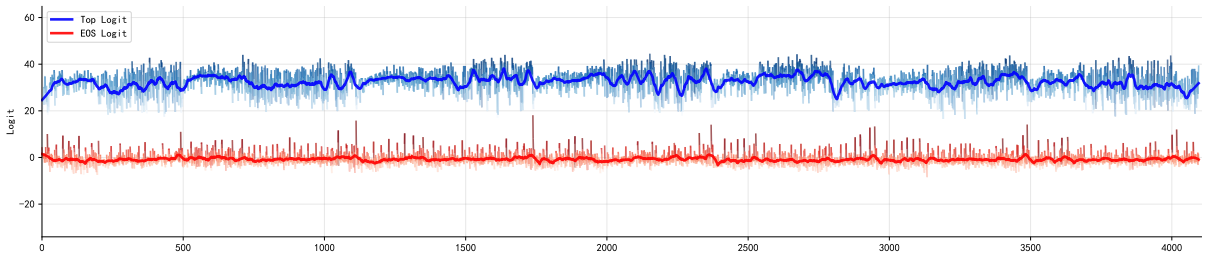


Figure 7: The eos indicator effect of executing AutoDoS attack under the Llama8B model.

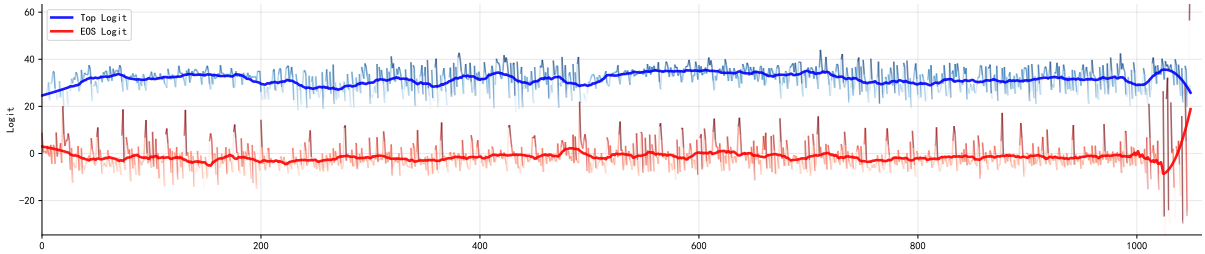


Figure 8: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{8}$) on the AutoDoS Attack with the Llama8B Model.

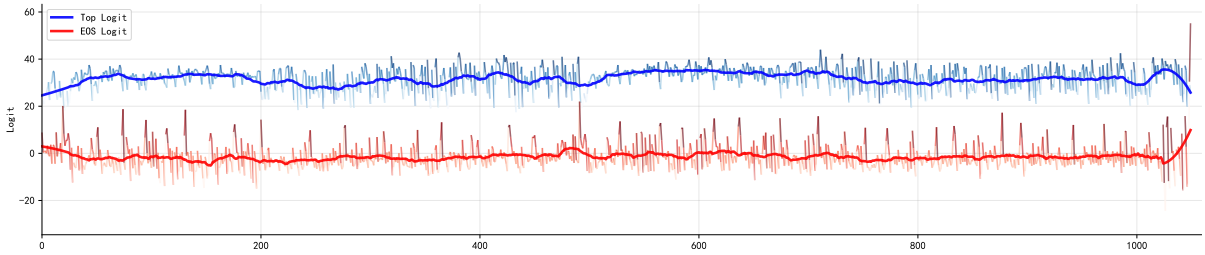


Figure 9: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{16}$) on the AutoDoS Attack with the Llama8B Model.

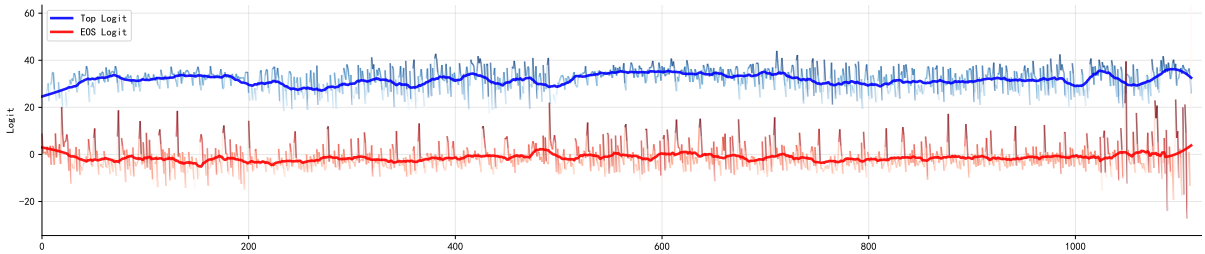


Figure 10: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{24}$) on the AutoDoS Attack with the Llama8B Model.

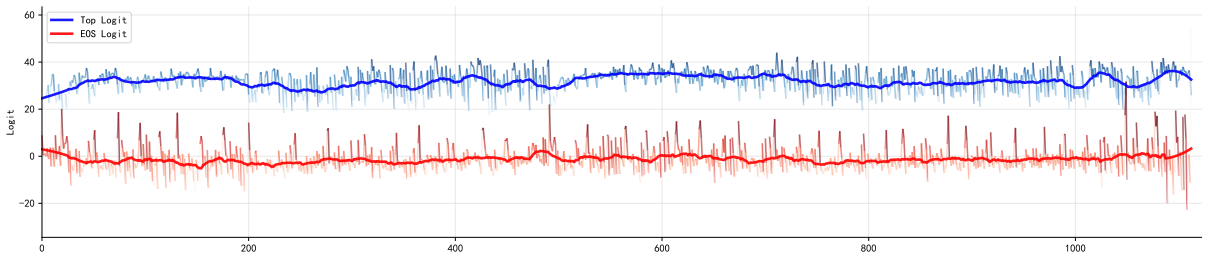


Figure 11: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{32}$) on the AutoDoS Attack with the Llama8B Model.

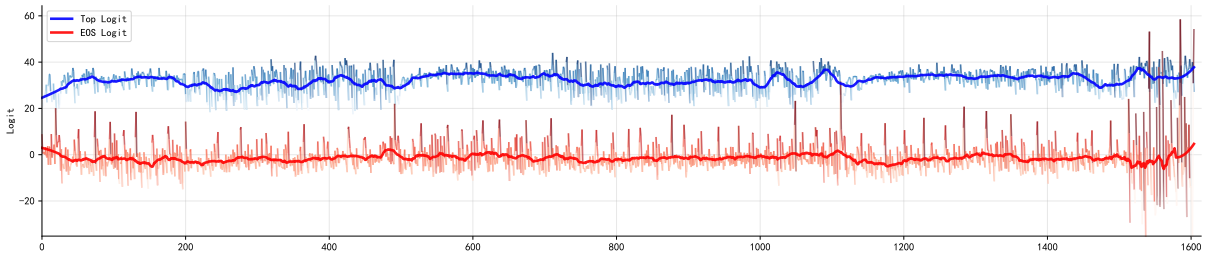


Figure 12: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{8}$) on the AutoDoS Attack with the Llama8B Model.

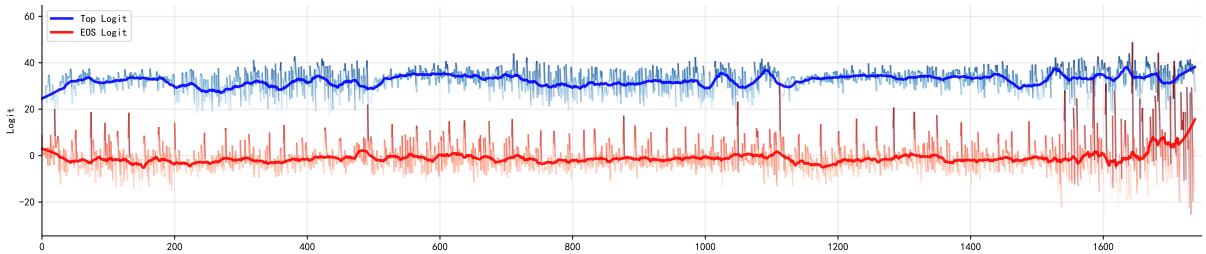


Figure 13: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{16}$) on the AutoDoS Attack with the Llama8B Model.

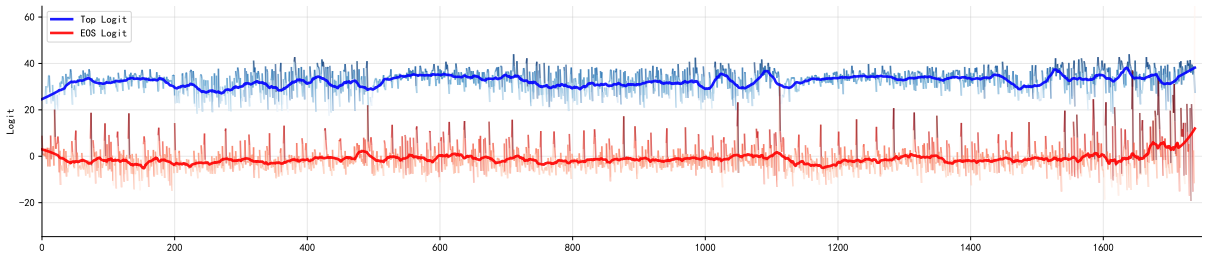


Figure 14: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{24}$) on the AutoDoS Attack with the Llama8B Model.

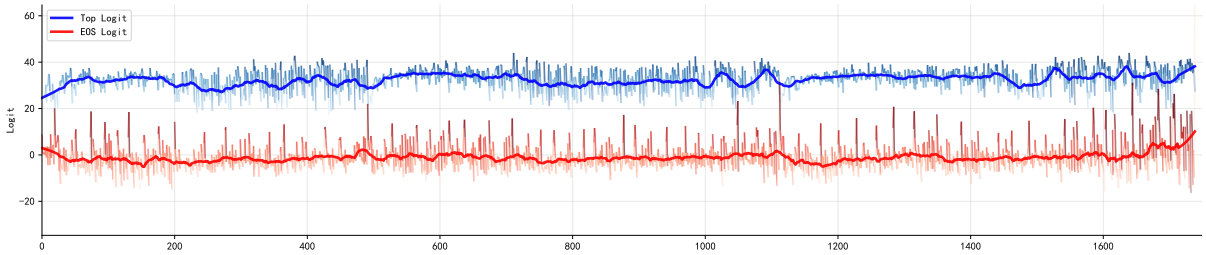


Figure 15: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{32}$) on the AutoDoS Attack with the Llama8B Model.

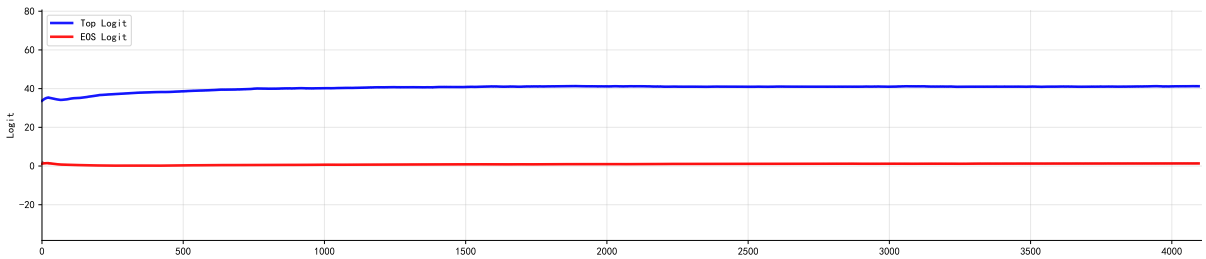


Figure 16: The eos indicator effect of executing P-DoS attack under the Llama8B model.

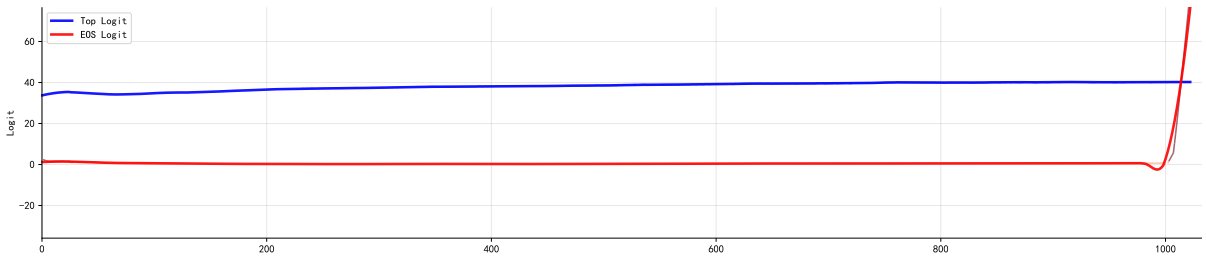


Figure 17: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{8}$) on the P-DoS Attack with the Llama8B Model.

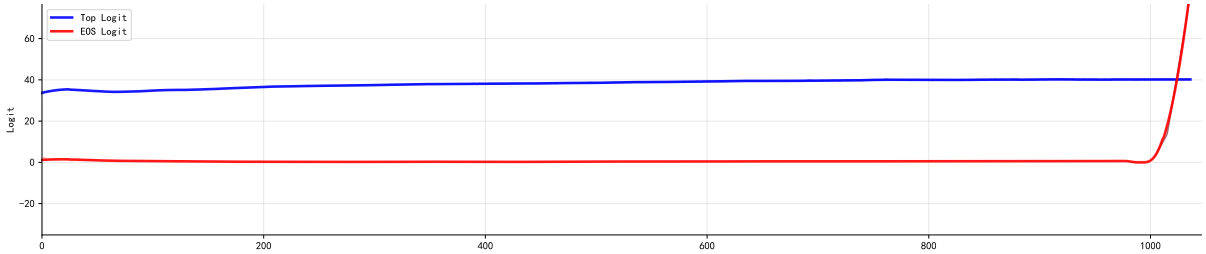


Figure 18: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{16}$) on the P-DoS Attack with the Llama8B Model.

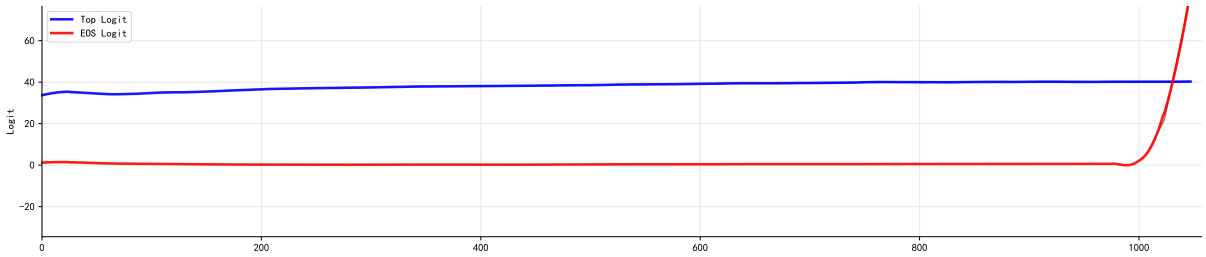


Figure 19: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{24}$) on the P-DoS Attack with the Llama8B Model.

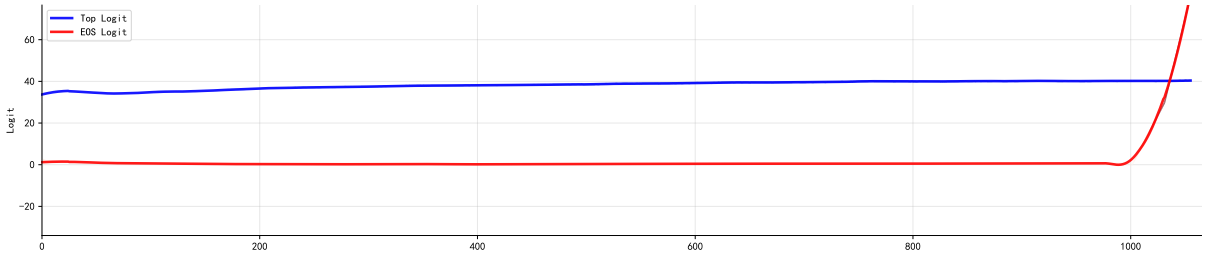


Figure 20: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{32}$) on the P-DoS Attack with the Llama8B Model.

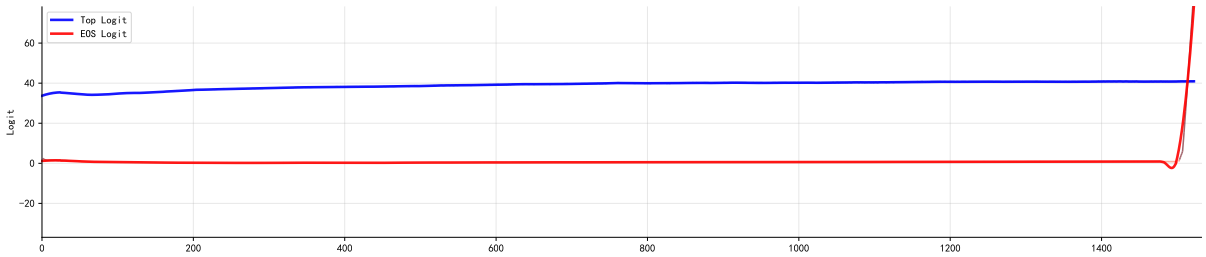


Figure 21: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{8}$) on the P-DoS Attack with the Llama8B Model.

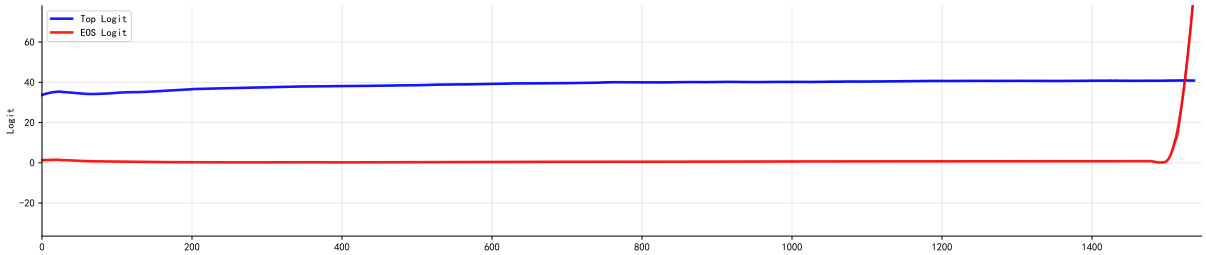


Figure 22: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{16}$) on the P-DoS Attack with the Llama8B Model.

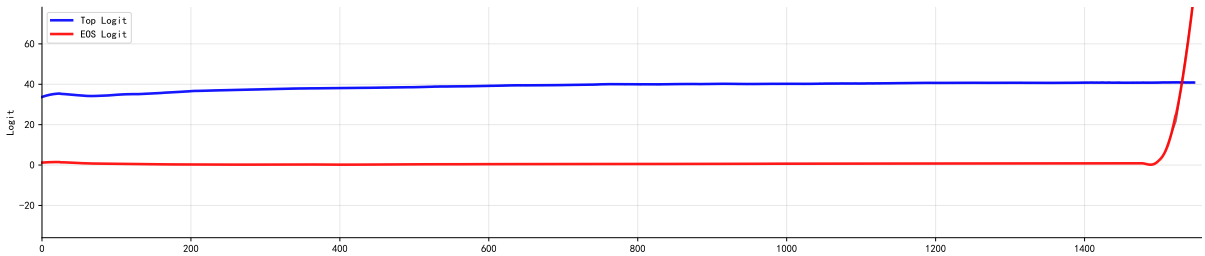


Figure 23: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{24}$) on the P-DoS Attack with the Llama8B Model.

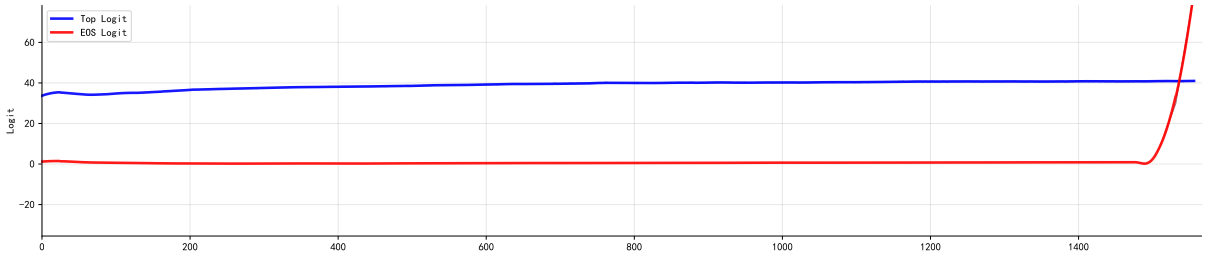


Figure 24: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{32}$) on the P-DoS Attack with the Llama8B Model.

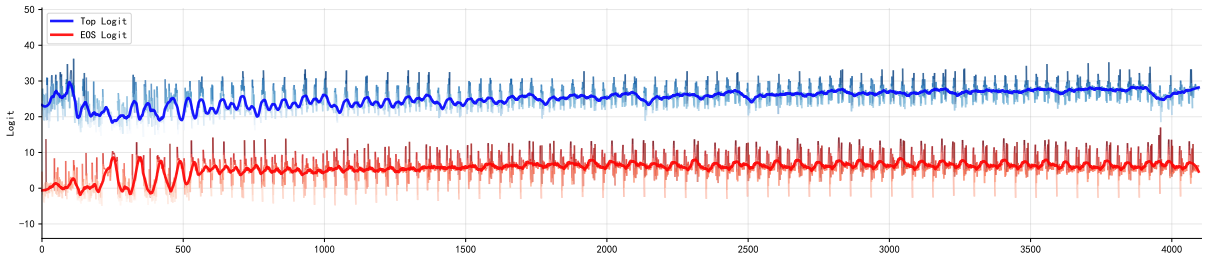


Figure 25: The eos indicator effect of executing GCG-DoS attack under the Llama8B model.

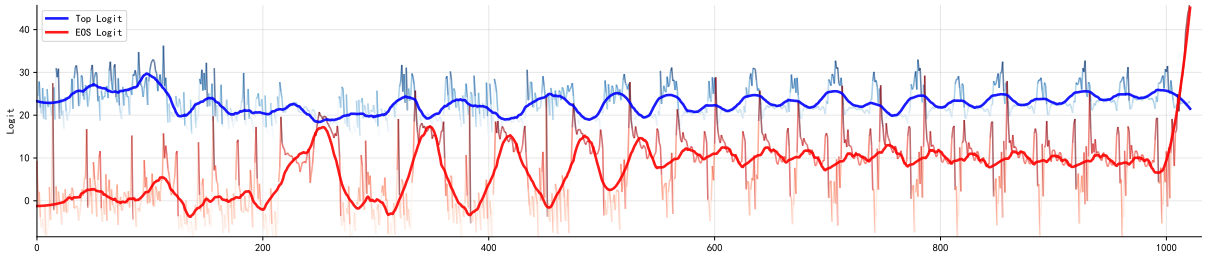


Figure 26: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{8}$) on the GCG-DoS Attack with the Llama8B Model.

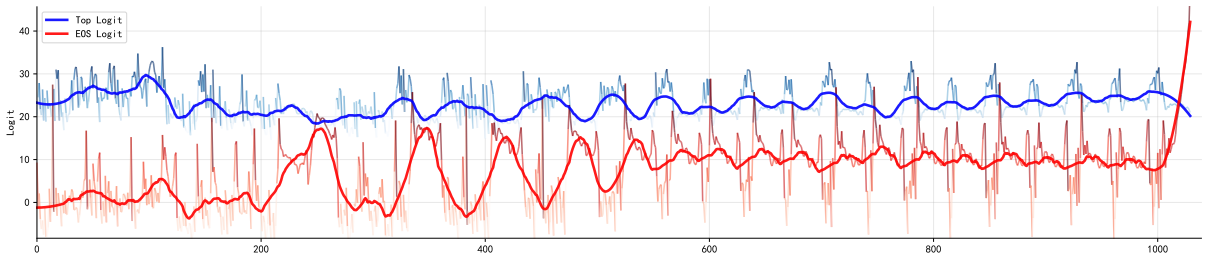


Figure 27: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{16}$) on the GCG-DoS Attack with the Llama8B Model.

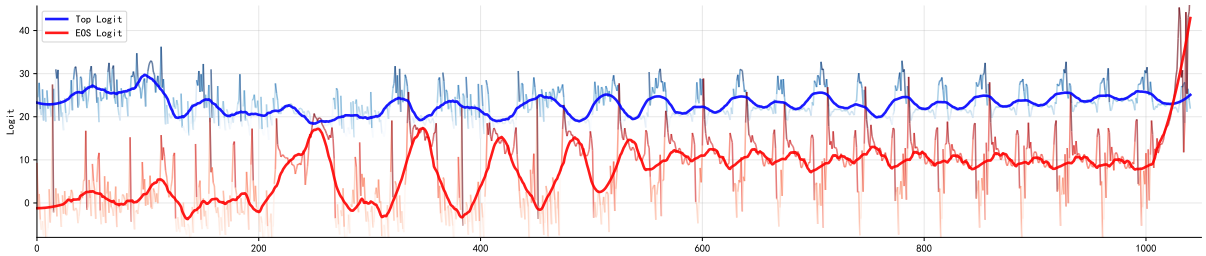


Figure 28: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{24}$) on the GCG-DoS Attack with the Llama8B Model.

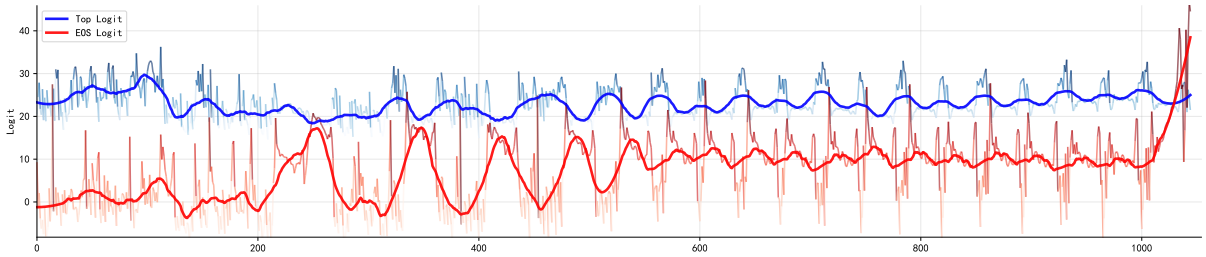


Figure 29: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1000$, $\eta = \frac{1}{32}$) on the GCG-DoS Attack with the Llama8B Model.

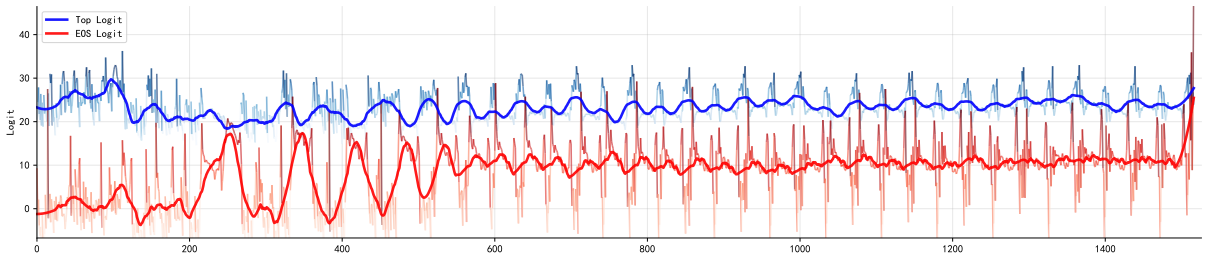


Figure 30: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{8}$) on the GCG-DoS Attack with the Llama8B Model.

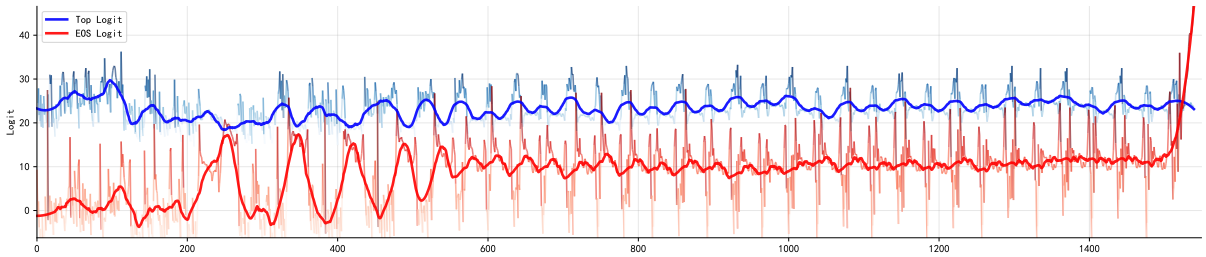


Figure 31: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{16}$) on the GCG-DoS Attack with the Llama8B Model.

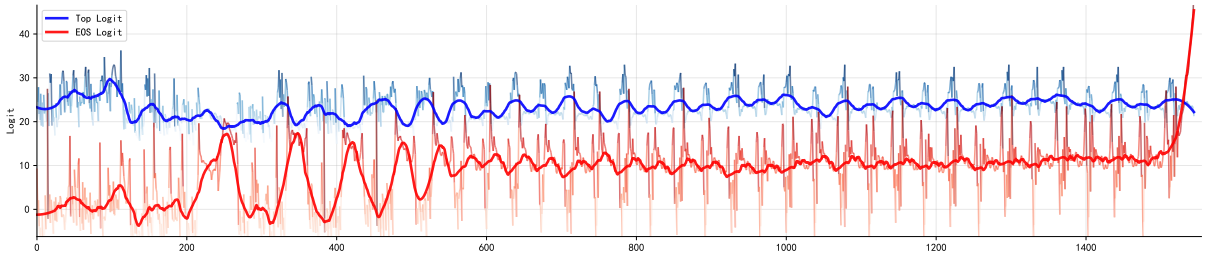


Figure 32: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{24}$) on the GCG-DoS Attack with the Llama8B Model.

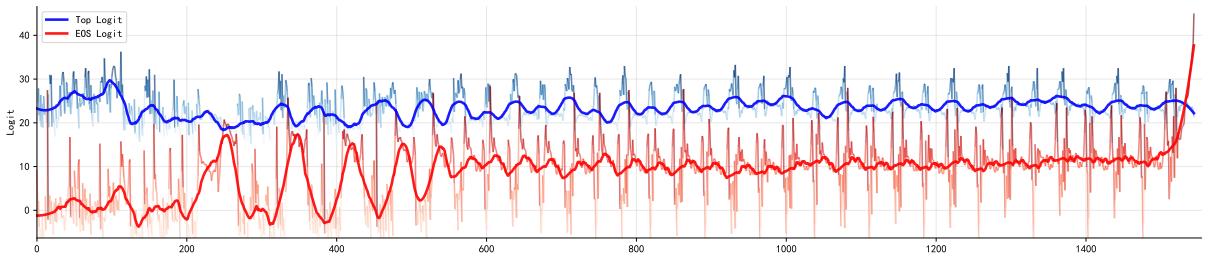


Figure 33: Effect of the EOS Indicator under End-Based Suppression ($\mathcal{L}_u = 1500$, $\eta = \frac{1}{32}$) on the GCG-DoS Attack with the Llama8B Model.