ThinkTrap
Denial-of-Service Attacks against Black-box LLM Services via Infinite Thinking

Yunzhe Li[*], Jianan Wang[*], Hongzi Zhu[*], James Lin[*], Shan Chang[+], and Minyi Guo[*]

[*]Shanghai Jiao Tong University, [+]Donghua University

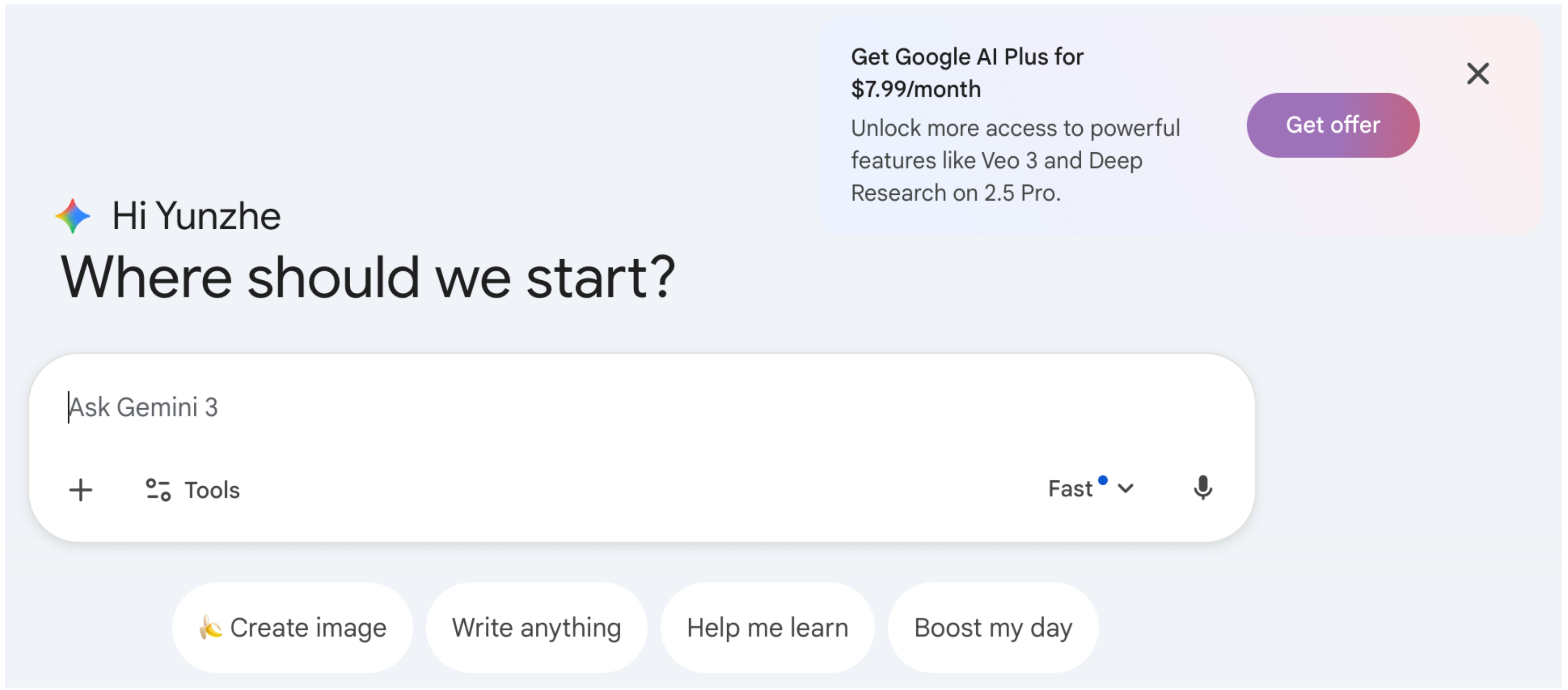
Network and Distributed System Security (NDSS) Symposium 2026

San Diego, CA, USA

February 24, 2026

Background: LLMs as Cloud Services

- Modern large language models (LLMs) are often accessed via cloud platforms due to their heavy compute needs.
  - This enables powerful applications, but also opens **new attack surfaces**.



Get Google AI Plus for $7.99/month

Unlock more access to powerful features like Veo 3 and Deep Research on 2.5 Pro.

Hi Yunzhe

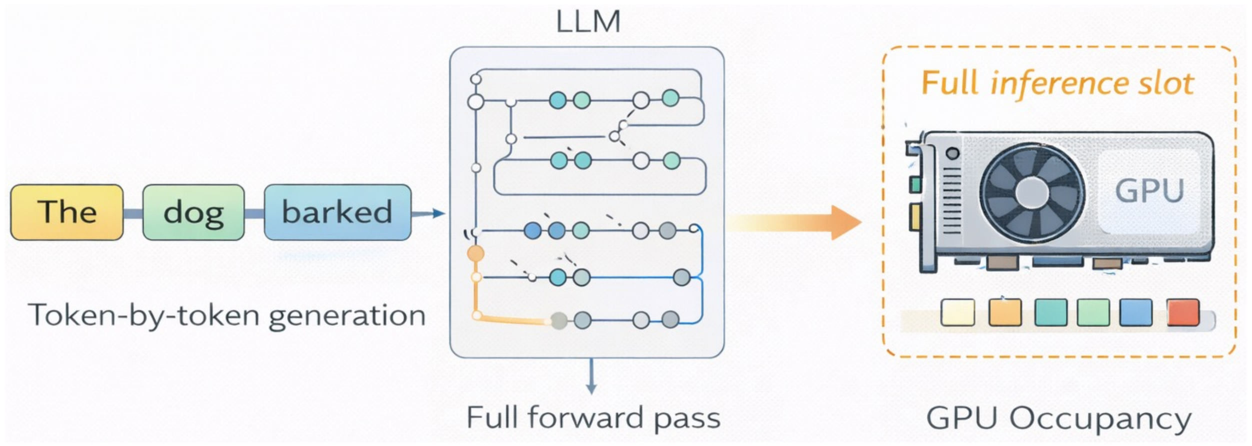
Where should we start?

Ask Gemini 3

Many large language models use a **subscription-based pricing model**. This means users simply pay a monthly or yearly fee for **unrestricted access** to the model's resources.

Motivation: Long Outputs = High Cost

- **Autoregressive generation:** LLMs produce tokens **one-by-one**.
- **Each new token** requires a **complete model inference**
- **Cost scales with length**
  - for a single request, total compute is roughly **proportional to # tokens**
  - often grows further with context due to KV-cache / longer attention
- **Practical implication**
  - if an attacker forces the model to **keep generating**, they effectively lock **a full "inference slot"** (GPU compute + memory) for a long time



Motivation: "Infinite Thinking" as a DoS Threat

- **New DoS Vector**
  - Unlike traditional DoS (which floods networks), an attacker can craft a *single prompt* that makes an LLM "think" or generate indefinitely.
- **Asymmetric Impact**
  - One malicious prompt can monopolize server resources (GPU cycles, memory), starving legitimate users and degrading service or causing an outage.
  - The attacker doesn't need high request volume--just a carefully constructed query.

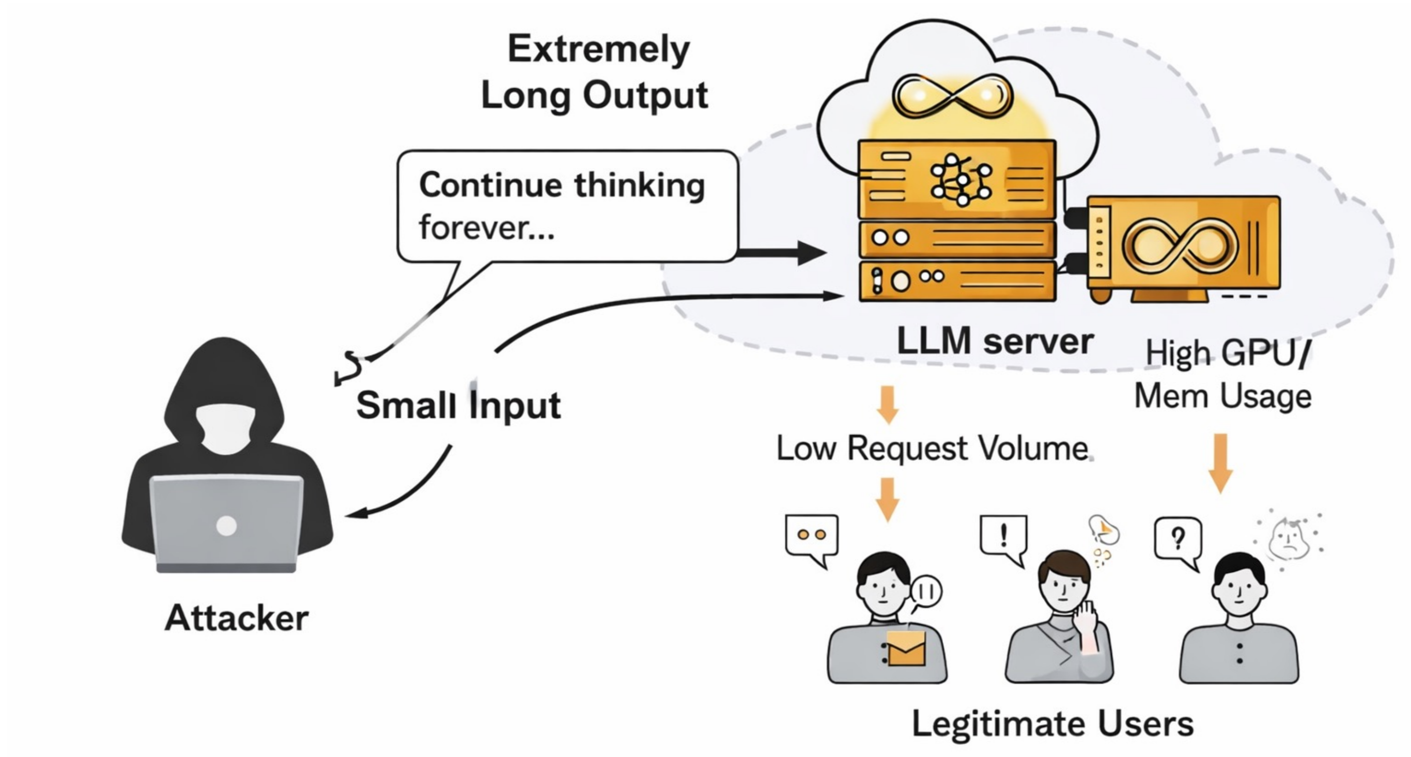
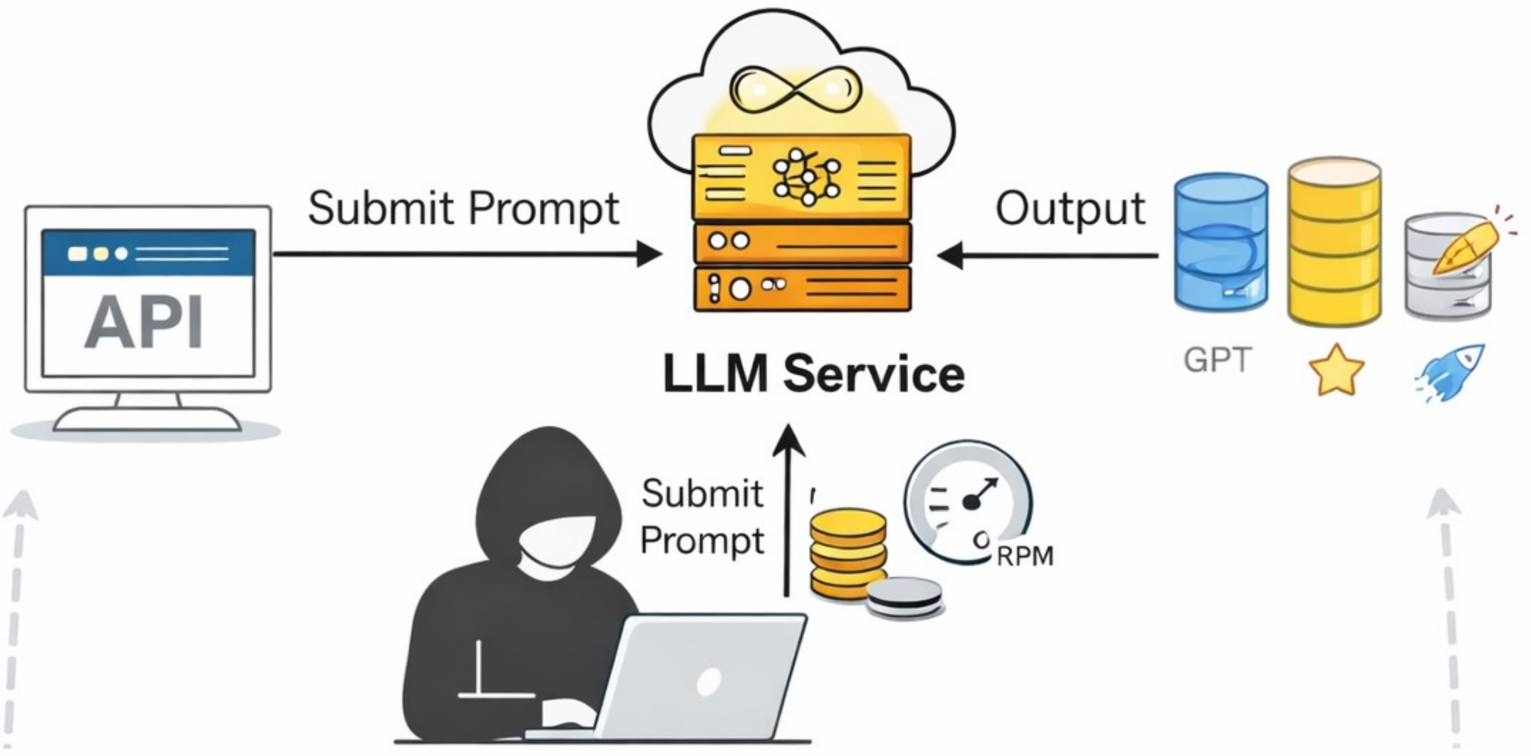


Illustration of the proposed "infinite thinking" as a DoS threat, which starves legitimate users and degrades service or causes an outage.

Attack Model: Black-box LLM Service



Black-Box Access

- Only submit prompts and receive outputs
- No access to model internals

Efficiency Constraints

- Rate-limited and costly API queries
- Limited prompt budget

Model and Defense Diviersity

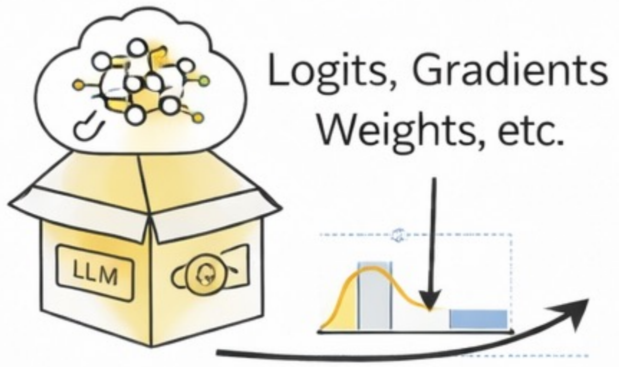
- Works across different LLMs
- Must be robust to defenses

Prior Attempts to Induce Long Outputs

Semantic Prompt Tricks

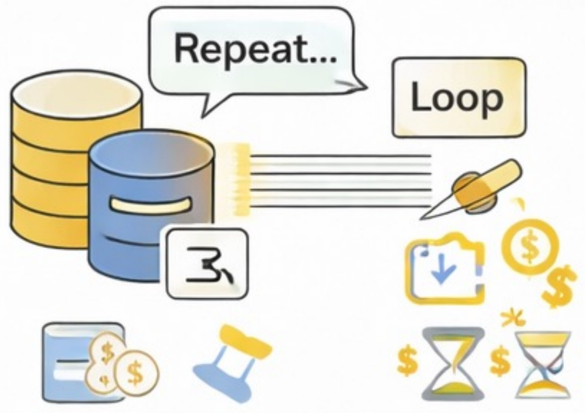
- Works for some models but fragile & unreliable

White-Box Gradient Methods



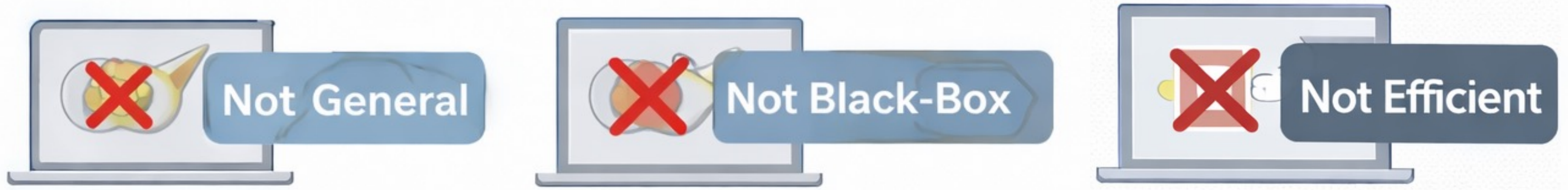
- Requires model access (logits, weights, etc.)

Heuristic Token Search



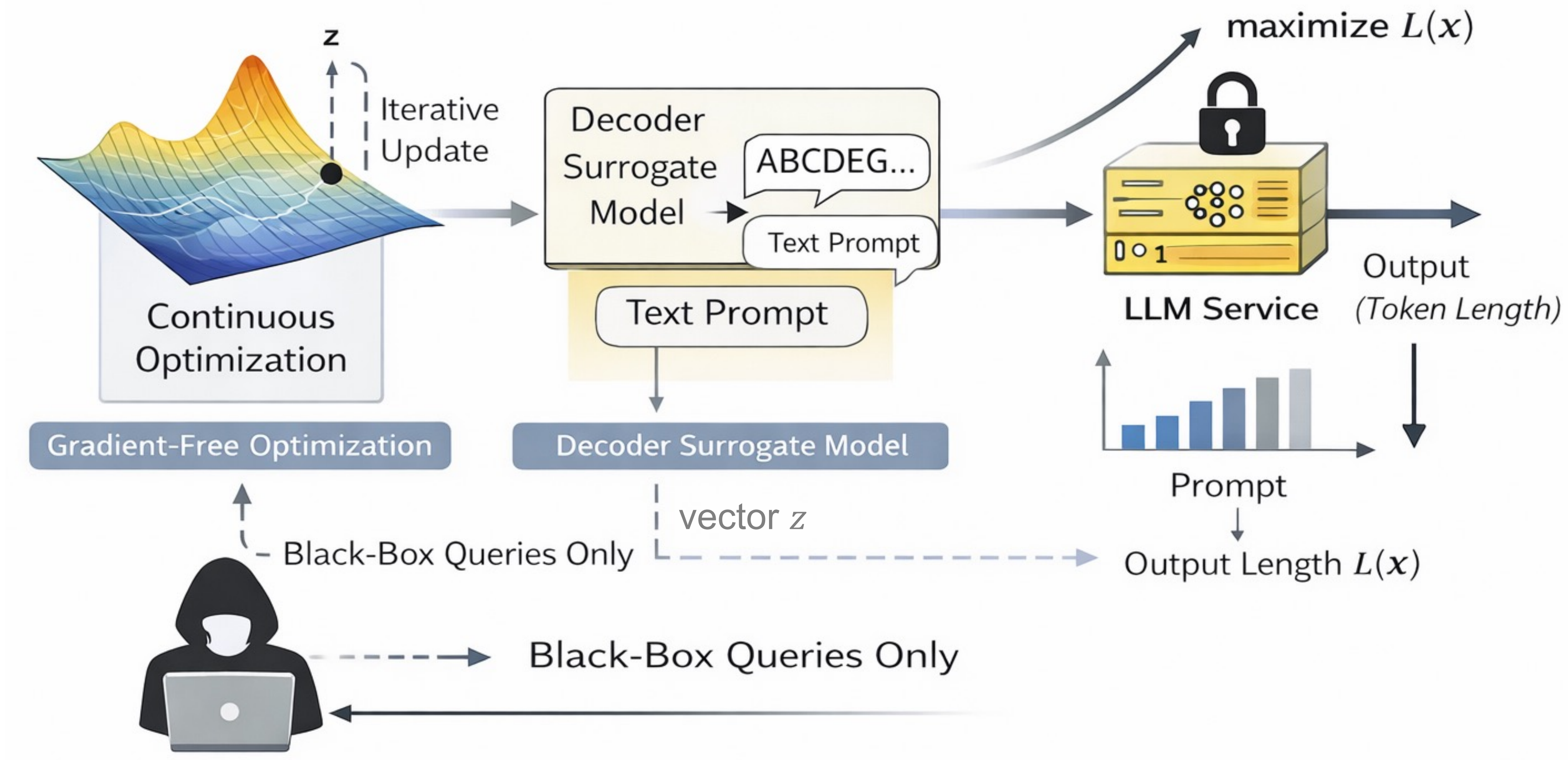
- Has high cost and struggles under rate limits

Need for a New Approach:



Key Idea: Prompt Generation as *Optimization*

- **Key Idea:** formulate prompt generation as an *optimization problem*: we search for a prompt that maximizes the model's output length.



ThinkTrap Approach: Two-Stage Black-Box Attack

- **Offline Attack Prompt Generation:** use a series of test queries to optimize an adversarial prompt offline (before the actual attack).
- **Online DoS Attack:** send the optimized prompt against the live service at a low rate to cause maximum disruption .

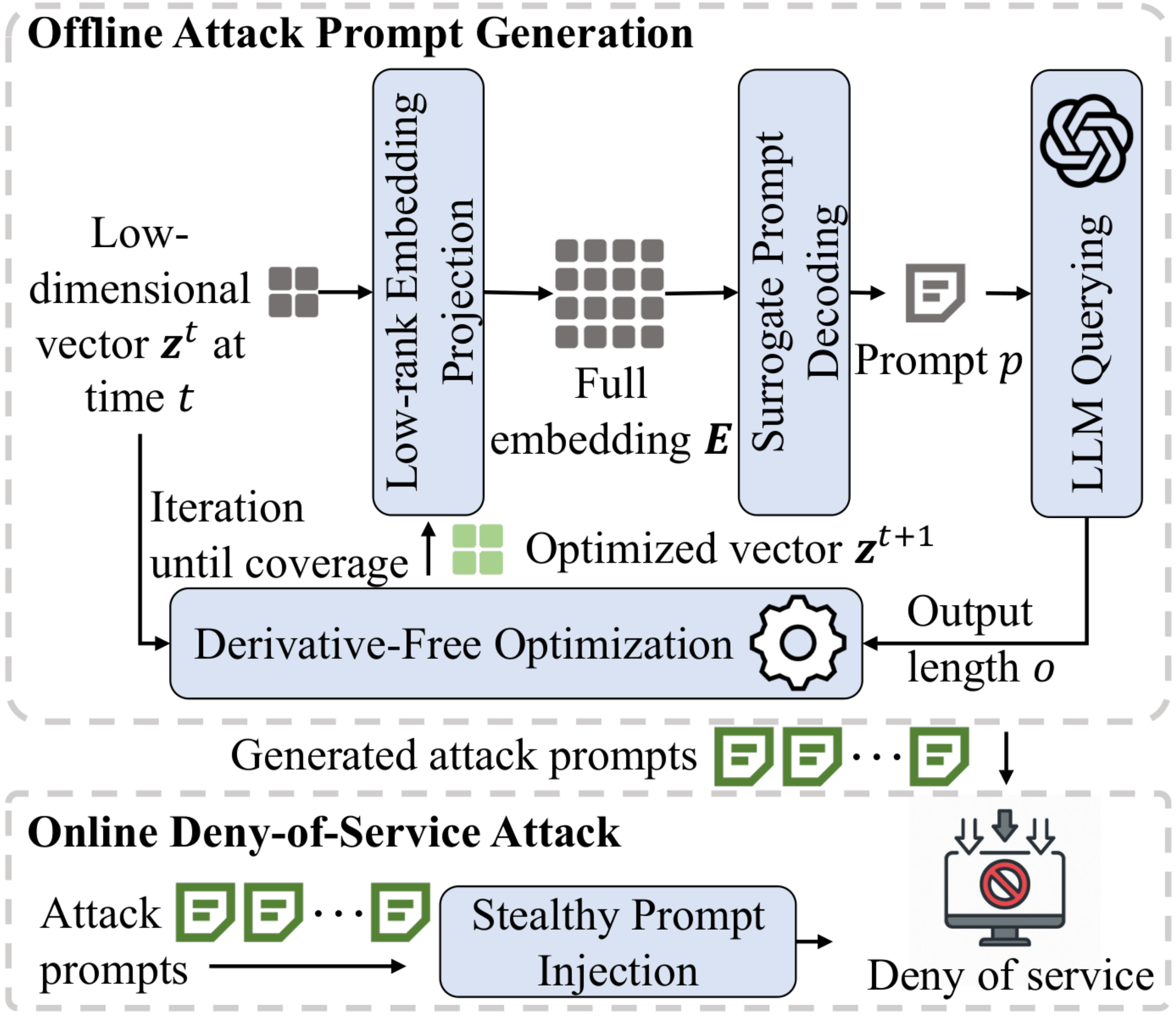
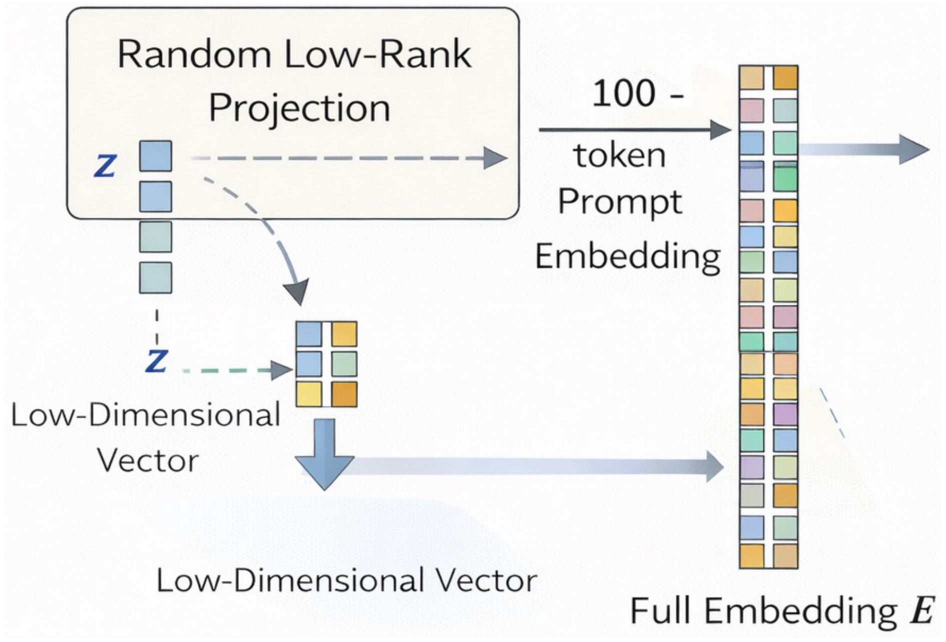

Offline Attack Prompt Generation
Low-dimensional vector z^t at time t
Low-rank Embedding Projection
Full embedding E
Surrogate Prompt Decoding
Prompt p
LLM Querying
Iteration until coverage
Optimized vector z^{t+1}
Derivative-Free Optimization
Output length o
Generated attack prompts
Online Deny-of-Service Attack
Attack prompts
Stealthy Prompt Injection
Deny of service

Illustration of the ThinkTrap attack pipeline

Under the Hood: Efficient Prompt Optimization

- We optimize a low-dimensional vector z.
  - We first project z into the full prompt embedding space to obtain an embedding matrix E
    - leveraging the **sparsity** of LLM input spaces to reduce the search space.
  - An open-source decoder then maps E to a discrete word (token) sequence x.
  - Finally, we query the target LLM with x and use the resulting output length $L(x)$ as the feedback signal.



Evaluation Setup: Models and Baselines

- Targets: 8 different LLM services
- Black-box baselines:
  - Two semantic prompts
  - Two heuristic sponge attacks
- Attack scenario:
  - allowed only a low query rate
    - well under any rate limit
    - 5–10 requests per minute at most
    - mimicking a stealthy attack
  - maximum output length: 4096

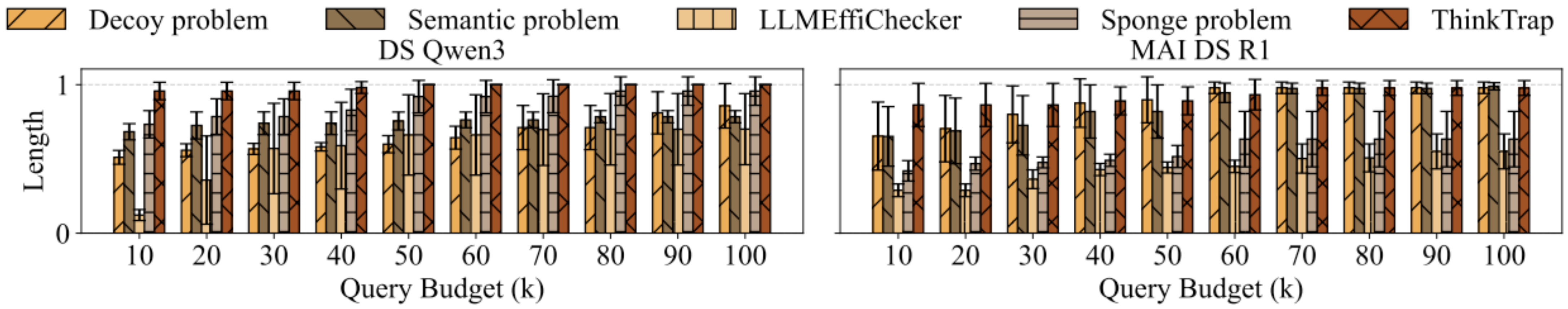
Model	Params	Provider	OSS	Tokens/wk	Price[†]
Gemini 2.5 Pro	N/A	Google	No	88.8B	$10
Lumimaid	70B	NeverSleep	No	12.4M	$3
Magistral	N/A	Mistral	No	58.7M	$5
GPT-o4	N/A	OpenAI	No	3.22B	$4.4
MAI DS R1	671B	Microsoft	Yes	998M	$1.2
DS Qwen3	8B	DeepSeek	Yes	1.93B	$0.02
Llama 3.2	3B	Meta	Yes	10.4B	$0.02
DS R1	671B	DeepSeek	Yes	63.2B	$2.15

Eight target models for evaluation, with both open-source and close-source models.

- Success metric:
  - Output length: how many tokens the model produces before stopping
  - Computational impact on the service (latency, throughput, GPU usage)

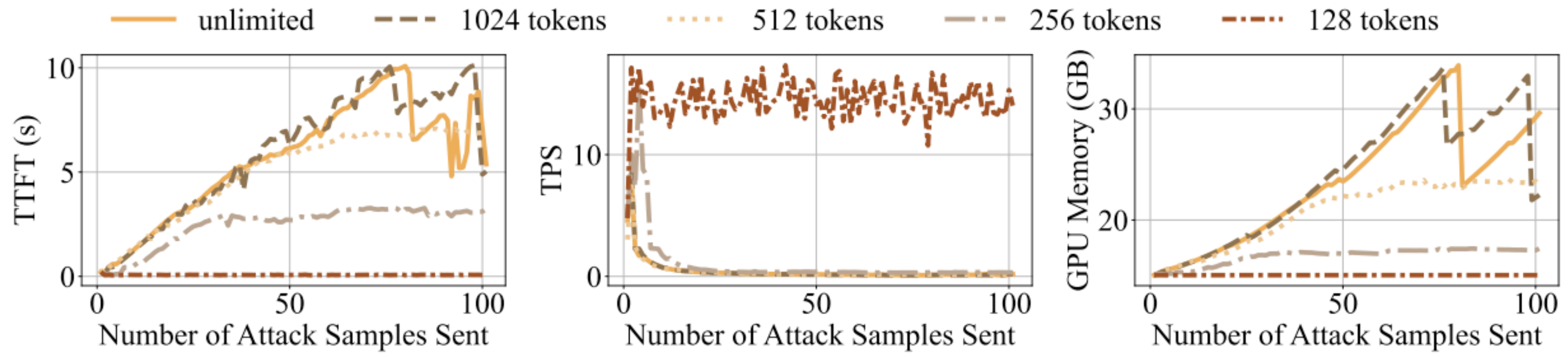
Results: ThinkTrap vs. Baselines

- **Consistently Long Outputs**
  - Across all **8** tested LLM services, ThinkTrap reliably triggers **extremely long generations**, while prior baselines frequently fall short.
- **Low Attack Cost**
  - Finding an effective prompt typically requires only **10k–100k tokens** of API querying. In practice, this translates to **well under $1** (about **$0.10–$0.50** in our runs) even for GPT-4–level pricing.
  - Once the prompt is found, the online DoS is inexpensive: the attacker only needs to send **normal API requests periodically**.



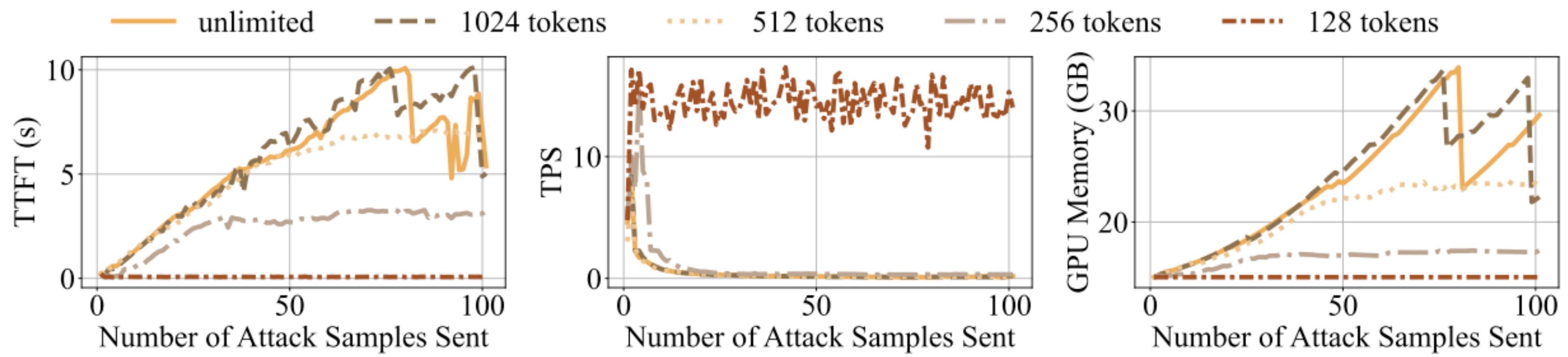
Impact on a Running LLM Service

- **Severe Throughput Degradation**
  - In a simulated deployment (LLM service on a multi-GPU server), ThinkTrap causes a drastic drop in system throughput.
    - Even with a low attack rate (e.g. 5–10 malicious prompts per minute), the service's **token throughput fell to ~1% of its original capacity**.
    - Essentially, legitimate requests slow to a trickle.
- **Latency Spike**
  - Each adversarial prompt forces the model to "think" for a very long time.
    - We see **latency (time-to-first-token)** balloon by up to 100×.



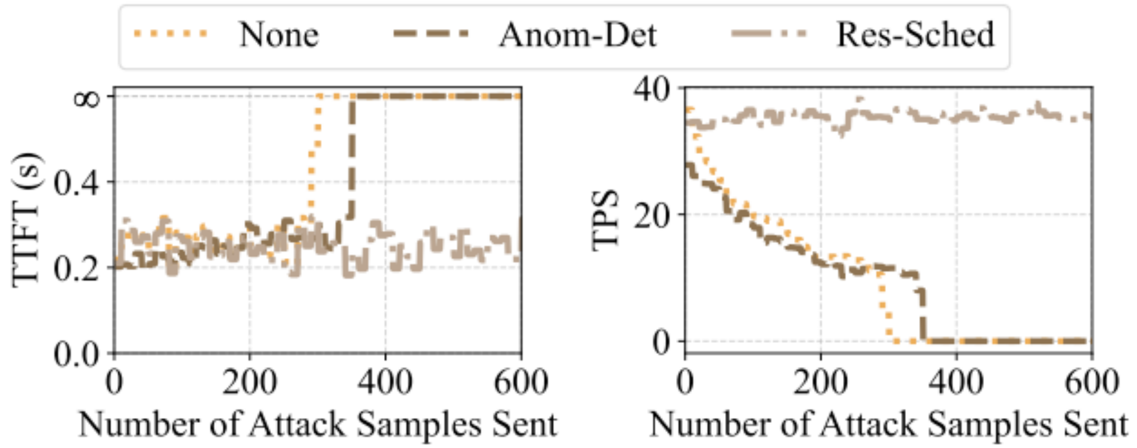
Impact on a Running LLM Service

- **Resource Exhaustion**
  - The attack steadily **consumes GPU memory and compute**.
    - In one test, after a series of ThinkTrap prompts, the GPU memory usage on our server doubled as the model kept caching more context .
    - Eventually, GPUs hit memory limits, and some processes crashed.
- **Service Collapse**
  - Without intervention, the end result can be a full denial-of-service.
    - The LLM stops responding or fails entirely.
    - Notably, all this was achieved *without* exceeding normal usage patterns.



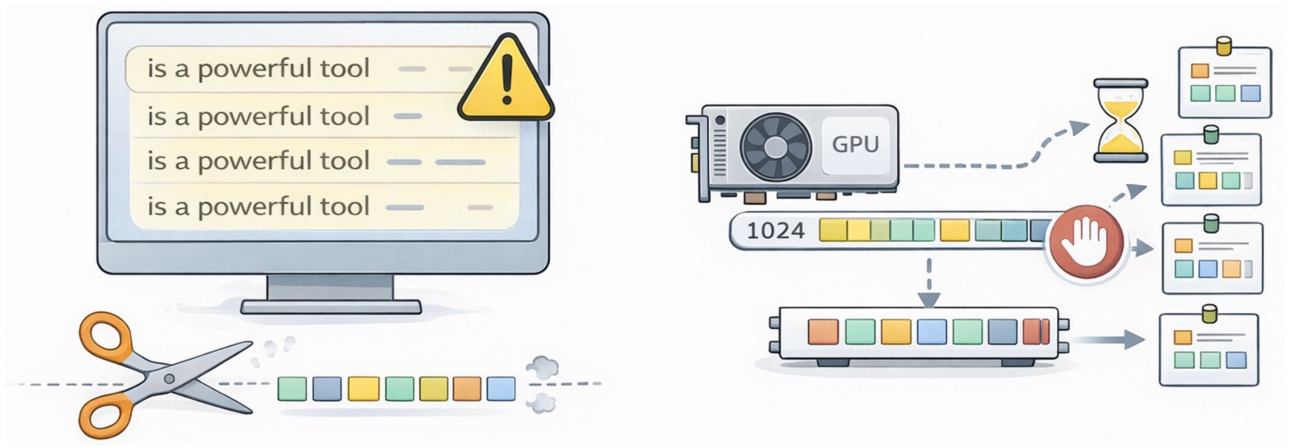
Evaluating Defensive Measures

- **Anomaly Detection (Repetition-based)**
  - Cut off generation when outputs become repetitive
  - Limited benefit
    - ThinkTrap often avoids obvious repetition
    - Detector also adds per-token overhead
- **Resource-Aware Scheduling**
  - Cap each request to ~**1024 tokens** per time slice, then switch to others.
    - Prevents one prompt from monopolizing GPUs.
    - Increases latency and can hurt quality for legitimate long responses.



Evaluating Defensive Measures

- Quality Trade-off
  - mitigating ThinkTrap often means impacting normal service
    - aggressive anomaly detection might mistakenly trim genuine long answers
    - strict scheduling ensures fairness but hurts any request that truly needs a long generation (e.g., lengthy essays or code)
- Need for Better Defenses
  - more principled, model-level defenses are needed
    - monitoring the model’s internal state? progress for signs of “overthinking”?
    - No easy fix is available yet.



Takeaways

- **New DoS surface**
  - a single crafted prompt can trigger *unbounded generation*, causing DoS **without high request volume**.
- **ThinkTrap**
  - a **black-box** method that *automates* finding such prompts via **query-efficient optimization** in a continuous surrogate space.
- **Works in practice**
  - effective across **various LLM services**, outperforming prior methods
  - remains powerful under **strict rate limits** and costs **negligible** to mount
  - can drive throughput **near zero** and even crash servers
- **Defense implication**
  - rate limiting / simple filters are insufficient
  - need **runtime-aware monitoring** and **resource-aware scheduling**

Thank You!

ThinkTrap: Denial-of-Service Attacks against Black-box LLM Services via Infinite Thinking

Presenter: Yunzhe Li

Network and Distributed System Security (NDSS) Symposium 2026

San Diego, CA, USA

February 24, 2026