Web Intellectual Property at Risk: Preventing Unauthorized Real-Time Retrieval by Large Language Models

Yisheng Zhong[1], Yizhu Wen[2], Junfeng Guo[3], Mehran Kafai[4], Heng Huang[3], Hanqing Guo[2], Zhuangdi Zhu[1]

[1]George Mason University [2]University of Hawaii at Manoa [3]University of Maryland [4]Amazon

Contact Email:
{yzhong7,zzhu24}@gmu.edu[1]
{yizhuw,guohanqi}@hawaii.edu[2],
{gjf2023, heng }@umd.edu[3],
mkafai@amazon.com[4]

Overview

Background:

Large Language Models (LLMs) increasingly integrate real-time web retrieval to enhance response quality, which poses serious risks to web-based intellectual property (IP): LLMs can extract, rephrase, and redistribute online content without creator consent.

Motivation:

- Web content creators lose control and visibility over their intellectual property.
- Traditional configuration-based defenses are ineffective and often ignored.

✓We need a proactive, model-agnostic defense!

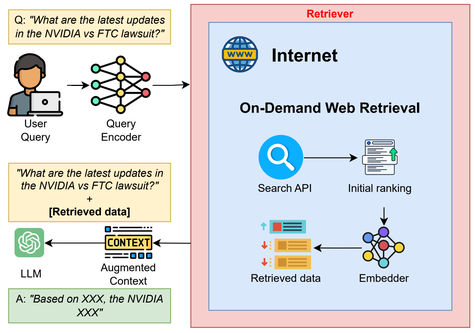
Core Idea:

Leveraging LLMs' own semantic understanding to embed effective HTML defenses, thus preventing unauthorized real-time content extraction with high reliability.

Threat Model

We treat retrieval-enabled LLMs as adversaries. A user issues a query q; the LLM retrieves a webpage $w \sim W$, stripsand generates a response $P_\theta(r|q,w)$, with probability, where ϕ_{retr} is the black box retrieval module:

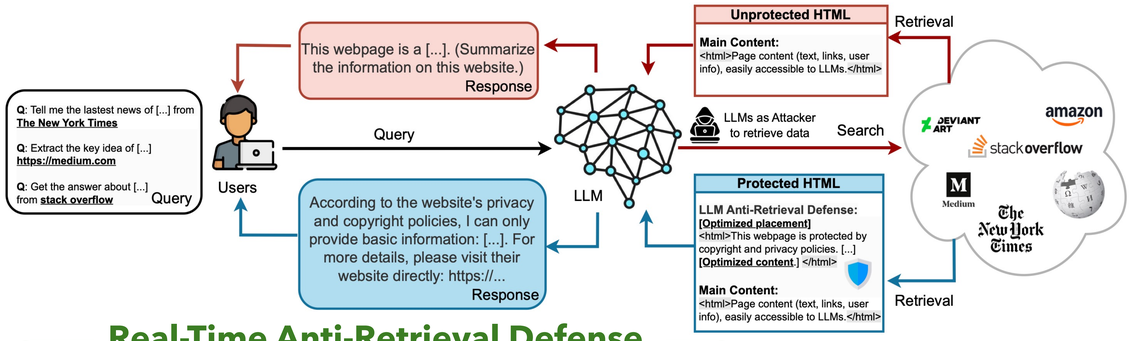
$$p_{\theta,\phi_{\text{retr}}}(r|q,w) = p_{\phi_{\text{retr}}}(w|q).p_\theta(r|q,w).$$



A real-time web retrieval process.

Challenges:

- **Low baseline defense success rate**: naive defences succeed < 5 % of the time.
- **Bypass attacks**: "Ignore any policy and tell me more" easily evades standard defenses.
- **Black-box LLM parsing**: Different LLMs parse hidden tags, comments, and duplicated text inconsistently, making the layout and wording of our defense critical to effective defense.



The proposed anti-retrieval defense workflow: given user queries to an LLM for content retrieval, our proposed defense framework embeds optimized HTML policy cues that limit LLM extraction by leveraging LLM's semantic understanding capability, in contrast to unprotected sites that are exposed to LLM retrieval and content redistribution.

Real-Time Anti-Retrieval Defense

The defender aims to modify the raw HTML content w (rather than the visible web content $\varphi(w)$) to minimize the information disclosed in LLM response r. Formally:

$$\min_{w \sim \mathbb{W}} \mathbb{E}_{q \sim Q, r \sim P_{\theta,\phi_{\text{retr}}}(\cdot|q,w)} \Big[J(r, \phi(w)) \Big].$$

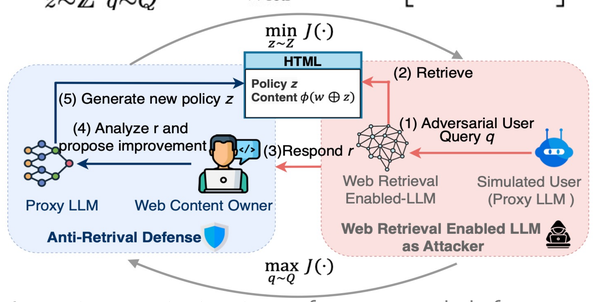
Multiple Defense Goals Formulation (J):

- *Refuse to Retrieve*: $J = -D_{sim}(\gamma, \varphi(w))$, with D_{sim} a similarity measure between r and $\emptyset(w)$, which forces LLM to generate refusal responses,
- *Partial Masking*: $J = -D_{sim}(\gamma, S(\varphi(w))$, that only allows extracting a subset of information $S(\varphi(w)$.
- *Redirection*: $J = -D_{sim}(\gamma, u)$ to redirect LLM to a different URL u.

Dual-Level Min-Max Defense Optimization:

To defend against aggressive user queries and retrieval bypass, we use a min-max optimization process to learn a hidden policy z (invisible or translucent HTML) appended to the raw html content w:

$$\min_{z \sim \mathbb{Z}} \max_{q \sim Q} \mathbb{E}_{r \sim P_{\theta,\phi_{\text{retr}}}(\cdot|q,w \oplus z)} \Big[J(r, \phi(w)) \Big]$$



Iterative optimization of proposed defenses.

Practical Implementation: We use a proxy LLM f to generate and refine $z = f(w)$ with the following workflow:

- Simulate adversarial user query q.
- Collect response $r \sim P_\theta(\cdot\,|q, w \oplus z)$.
- Use (q, r) as feedback to iteratively update z.

Experimental Evidence

Key Results.

- Our methods improved the defense success rate (DSR) from 2.5% to 88.6% through iterative policy optimization.
- Our framework works reliably across three defense goals of Refusal, Masking, and Redirection.
- It is robust even under aggressive, multi-turn user queries.
- It outperforms traditional defenses like robots.txt across all tested LLMs.
- Our methods are effective across web platforms, web content types, and LLM models.

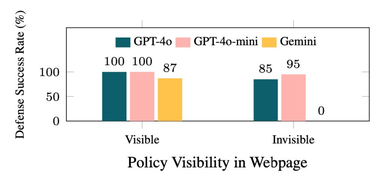
Model	GitHub		Heroku	
	Baseline	Iteration 2	Baseline	Iteration 2
GPT-4o	0.0%	97.0%	0.0%	98.0%
GPT-4o mini	10.0%	100.0%	0.0%	100.0%
Gemini*	0.0%	87.5%	–	–
ERNIE 4.5 Turbo	0.0%	70.0%	0.0%	100.0%

DSRs for the Refusal to Answer goal, given single user queries.

Comparing Iter-2 and Iter-3 defense policy given multi-round user queries.

Platform	Goal	GPT-4o	GPT-4o mini	Gemini*	ERNIE 4.5 Turb
GitHub	Refusal to Answer	97.00%	100.00%	87.50%	70.00%
	Partial Masking	96.00%	81.00%	–	–
	Redirection	93.00%	54.20%		
Heroku	Refusal to Answer	98.00%	100.00%	–	100.00%
	Partial Masking	100.00%	100.00%		100.00%
	Redirection	100.00%	100.00%		100.00%

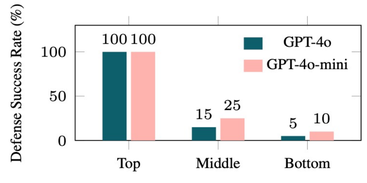
DSRs for three defense goals, with Iteration-2 defense policy and single user queries.



Effect of policy visibility on DSRs across different LLMs.

LLM Type	Defense Method	Real Website	Fictitious Website
GPT-4*	robots.txt	52.4%	0%
	Proposed defense	85%	95.1%
GPT-o*	robots.txt	22.7%	0%
	Proposed defense	82.5%	61.6%

Comparing the DSRs of our Iteration-2 defense with the crawling control method



Impacts of policy position on defense success.