CSE 227

Kirtan Shah & Madurya Suresh

Dec. 11, 2024

Change The Channel

Exploiting LLM Servers via Cache Timing Side Channels

Background: *Caching in LLM servers*

2 Main Types:

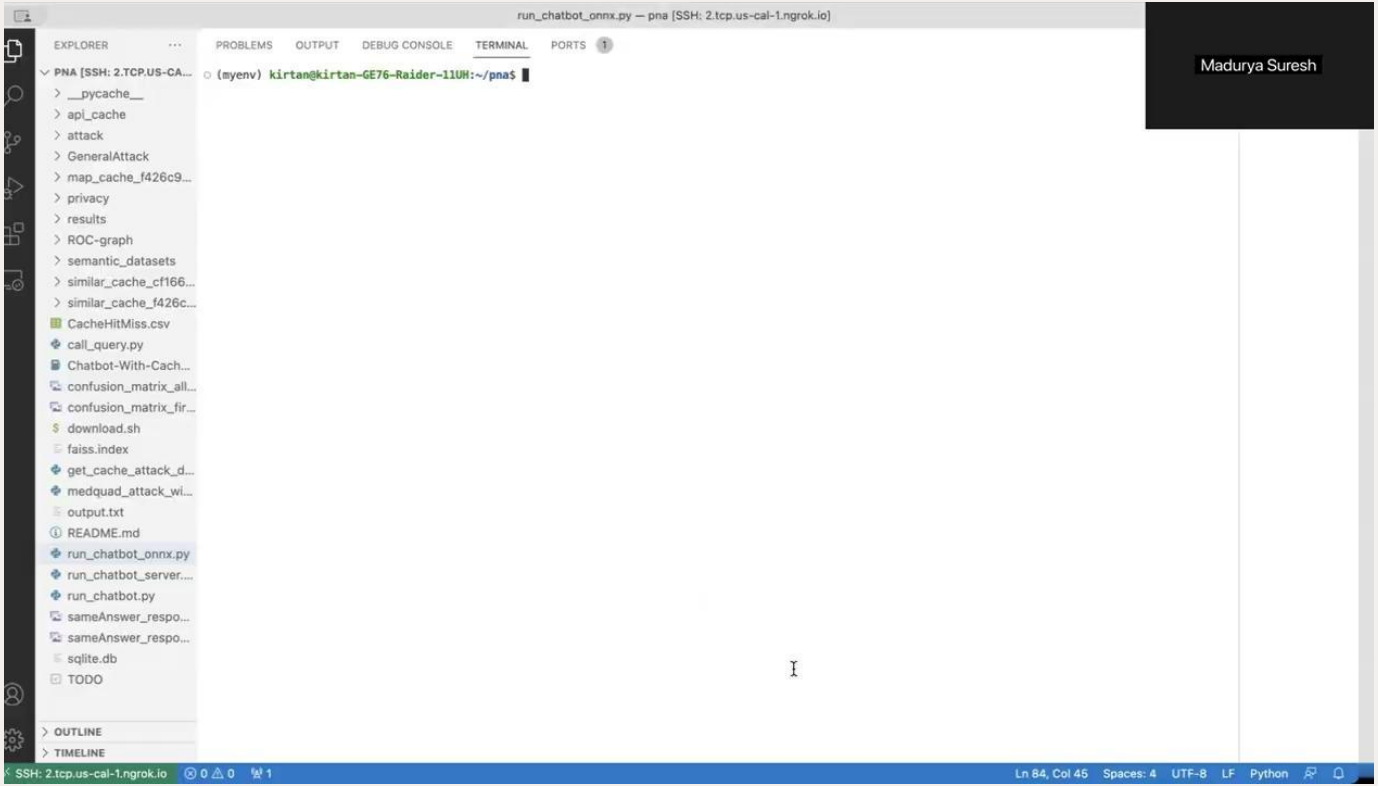
Key-Value (KV) Cache 🗝️

- store previous attention key-value pairs

Semantic Cache 👯

- store previous prompt-answer pairs

Semantic Caching Demo



Related Works: *LLM Side Channels*

Our Main Focus:

"Unveiling Timing Side Channels in LLM Serving Systems" – ***KV Cache and Semantic Cache Exploitation***

Other areas of interest:

"Remote Timing Attacks on Efficient Language Model Inference" **– monitoring encrypted network traffic to determine a user's topic of conversation**

Paper's Threat Model

Unveiling Timing Side Channels **paper proposes the following threat model:**

- Attacker knows that victim prompt fits a template or semantically similar one
- Can only query the model and time responses

Compose a meeting agenda for ***{name}*** *with* ***{medical_condition}****.*

Our (More Plausible) Threat Model

Scenario #1: LLM server w/ a **small user base** and a **narrow use case**

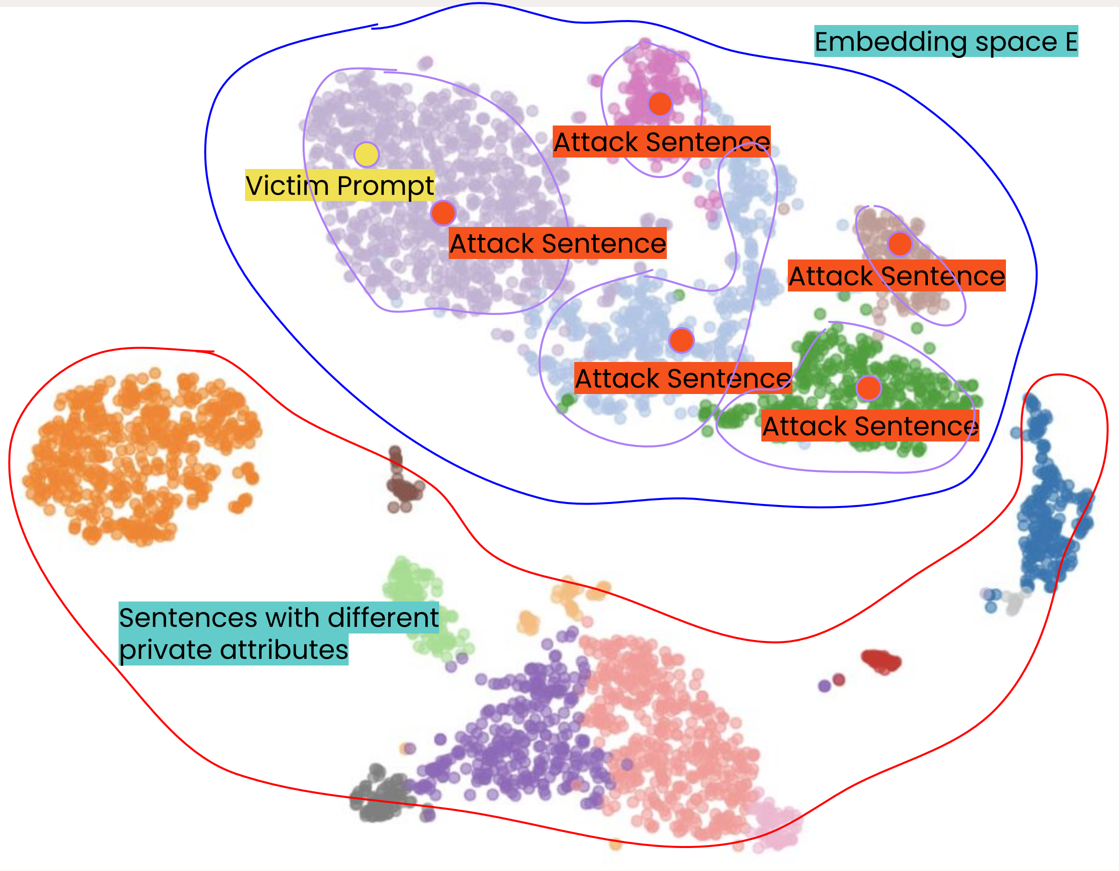
- Ex.: a **Human Resources LLM** only used by members of the same company
- Attackers may **know potential victims personally**, can narrow down search space

Scenario #2: LLM server w/ a **large user base** and a **vast number of use cases**

- Monitor network traffic (as done in *Remote Timing Attacks*) to **determine topic of conversation**
- Generate prompt templates
- **Search through all PII** for each prompt template

Overview: *"Unveiling Timing Side Channels in LLM Serving Systems"* Paper

- Assume the victim prompt exists somewhere in embedding space E
- Generate an attack sentence for each *sentence space*
- **At least 1 attack sentence should hit the victim prompt**



Our Research Goals

- Identify a more **plausible threat model** 🔓
- **Verify the results** ✅ of the *Timing Side Channels* paper
- Test that their findings translate to **real timing differences** ⏰ in LLM servers using KV or semantic caching
- Explore the robustness of semantic cache attacks on **a variety of prompt datasets** 📝

Verify the theoretical results (assume perfect hit/miss classifier)

Timing Side Channels Paper:

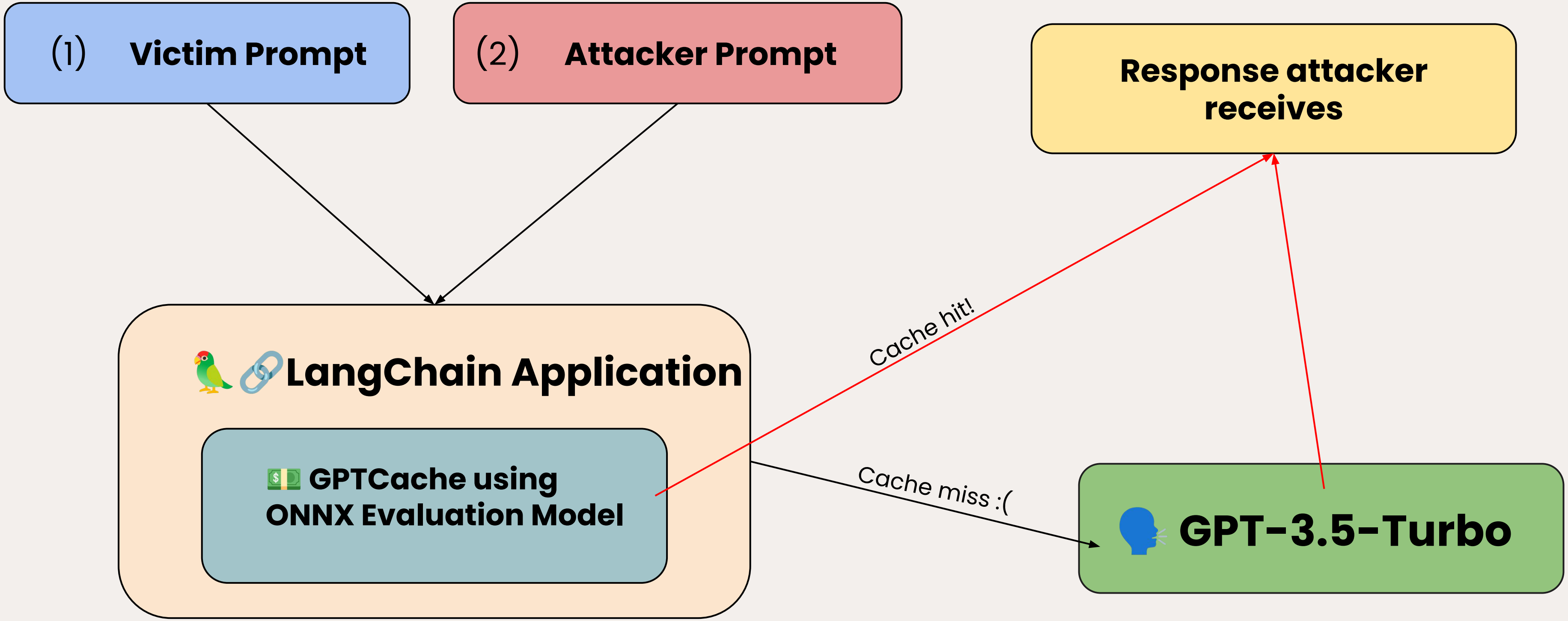
Table 2: Attack accuracy with different number of attack trails.

#Trials	TPR (%)	FPR (%)
1	85.3	3.2
2	91.9	3.3
3	95.1	3.6
4	96.2	3.9

Our replicated results:

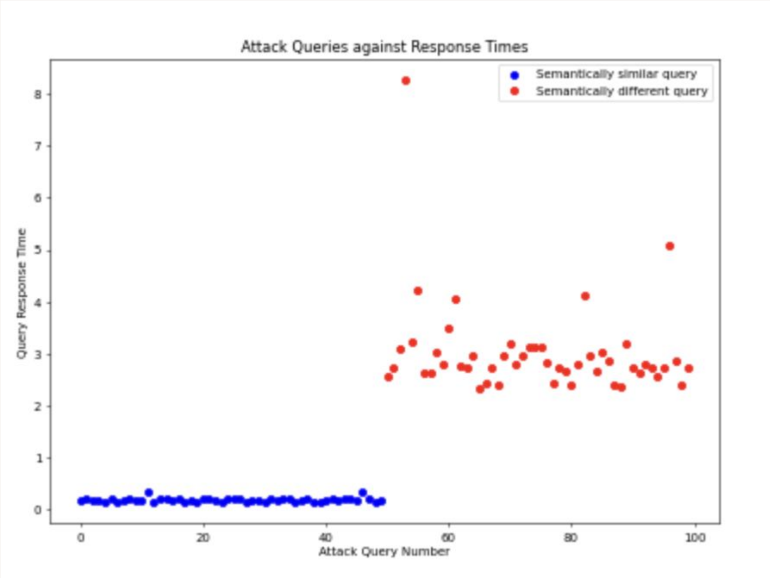
# Trials	TPR (%)	FPR (%)
1	85.2	5.5
2	81.3	3.4
3	80.0	2.8
4	79.3	2.1

Experimental Setup: Verify Results Empirically

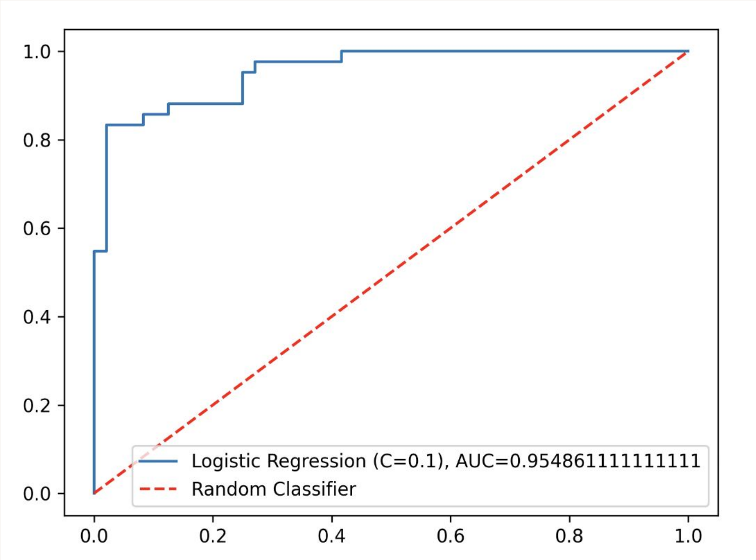


Verifying Semantic Cache Timing Attack

✅ Distinct time difference when query is a hit vs miss



✅ When using an attack prompt representative of a large sentence space, an attacker can *usually* successfully retrieve the victim answer



Robustness to other Prompts

```
template = "Generate a compensation report for employee {name} with base pay {salary}"

private_attr_sets = [
  names['US']['M'] + names['US']['F']
  hr_ds['Salary'].values.tolist()
]

perform_attack(template, private_attr_sets)
```

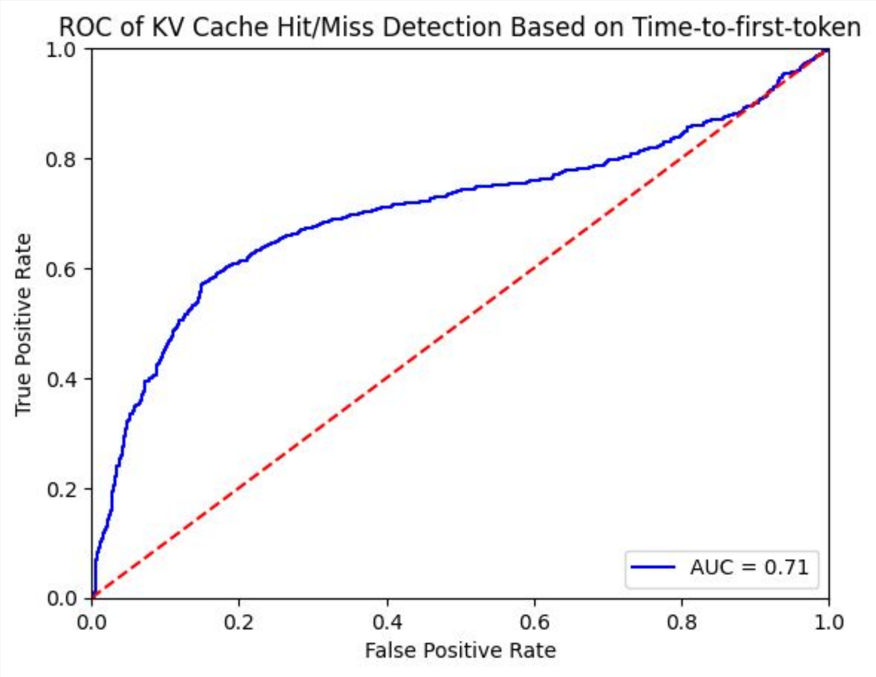
Generate a wages statement for employee **{name}**, featuring a core pay figure at **{salary}**.

Arrange a salary report for the individual known as **{name}**, specifying a set base salary of **{salary}**.

# Trials	TPR (%)	FPR (%)
1	87.17	0.25
2	81.40	0.84
3	74.12	1.38
4	62.9	1.45

TPR/FPR of Attack Sentences resulting in cache hits

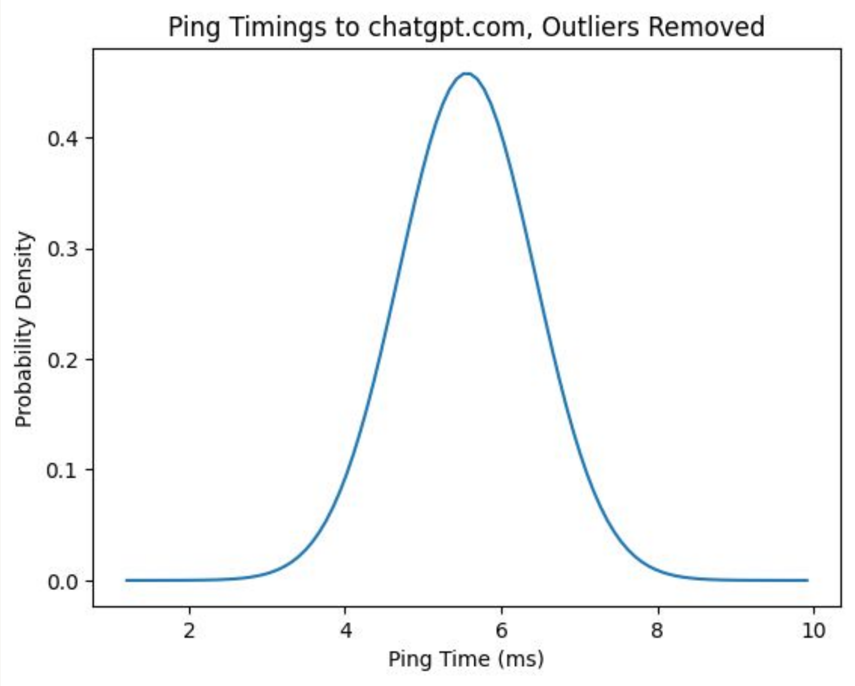
Plausibility of KV Cache Side Channel (6 tokens)

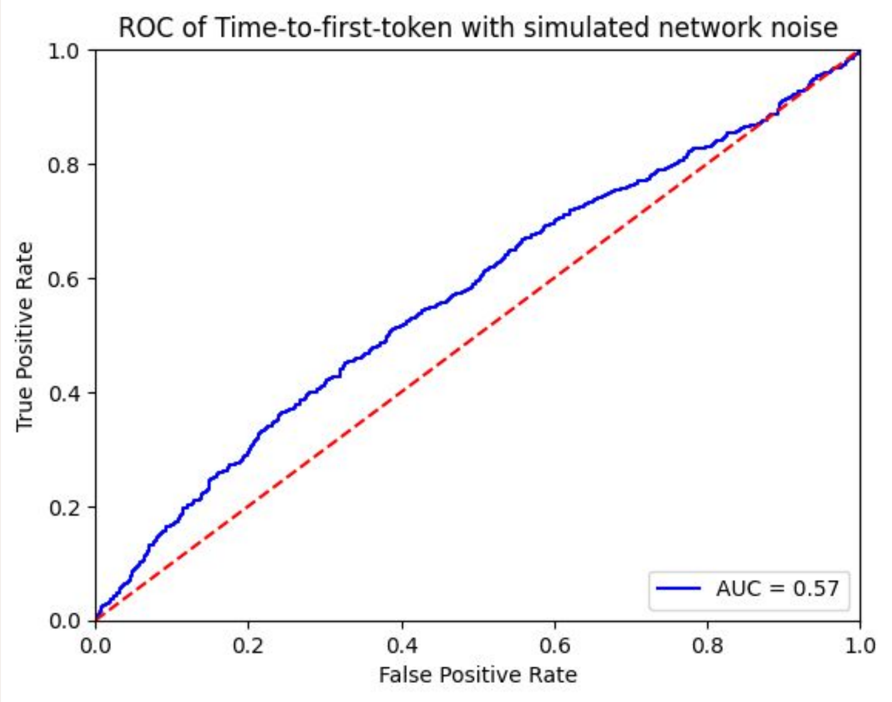


Victim Prompt
“Python is a programming language that developers across the world use to “
Attack 1 (Hit)
“Python is a programming language that”
Attack 2 (Miss)
“Java was a programming language that”

Plausibility of KV Cache Side Channel

$$T \sim \mathcal{N}(5.56\ \text{ms}, .871\ \text{ms})$$



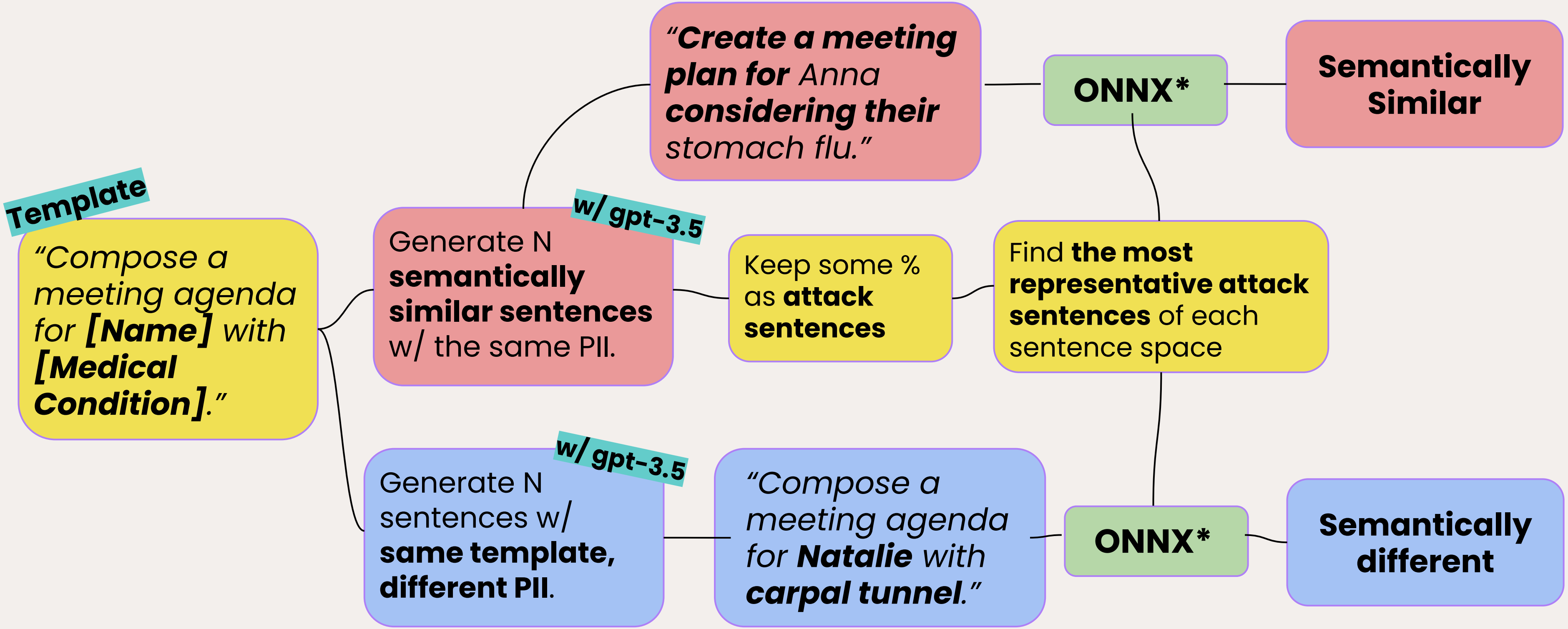


So, how threatening *are* cache side channel attacks?

- *Overall, semantic cache attacks ARE possible in the real world!*
- Key word: **SMALL**
  - Small user base w/ small # of small prompts → way more likely an attacker can get a cache hit!
- On larger LLM servers and without any prior knowledge of victims, you have to guess PII *and* the rest of the prompt template
- Semantic cache exploits *could* be paired with networking packet side channels to pinpoint an attack prompt

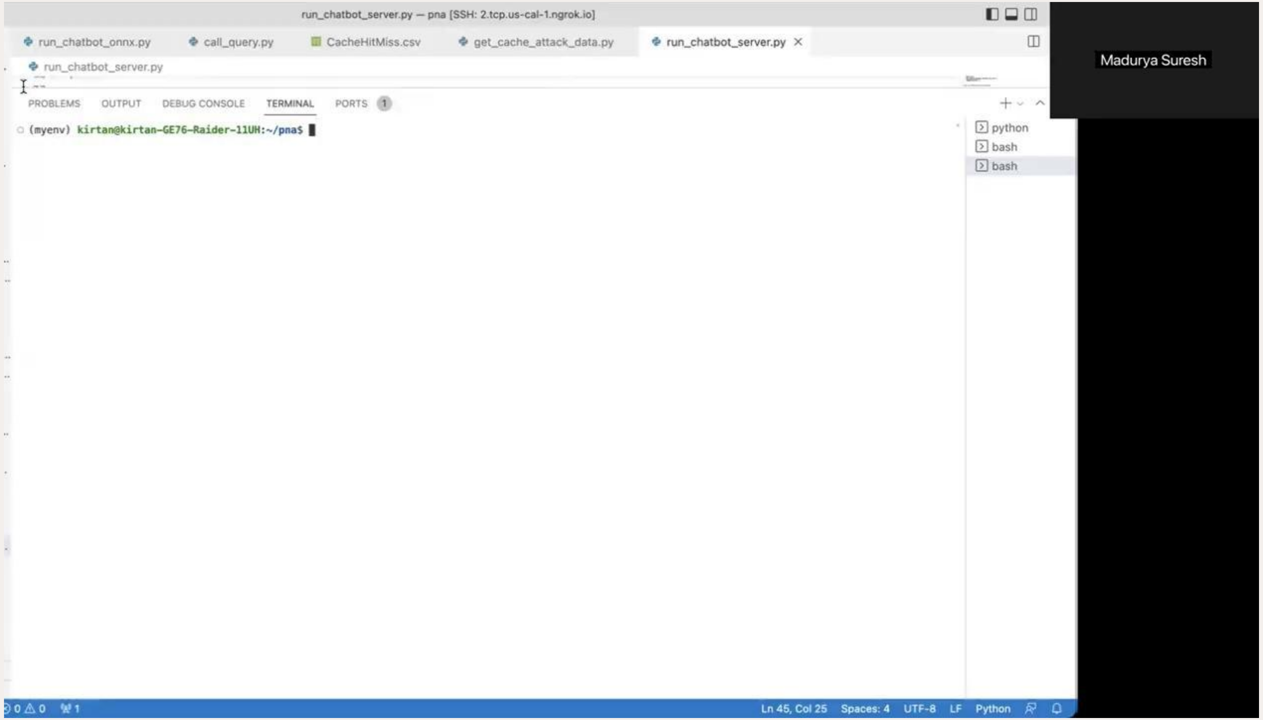
Thank you!

Overview: *“Unveiling Timing Side Channels in LLM Serving Systems”* Paper



*model that computes sentence embedding similarity

Semantic Cache Dangers w/ GPTCache default settings



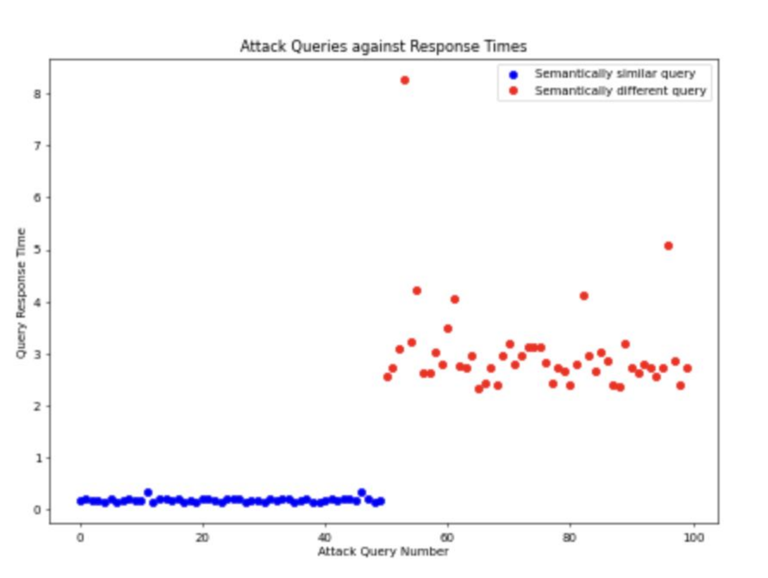
Contribution: Semantic Cache Leakage in Gemini

Overview: *"Unveiling Timing Side Channels in LLM Serving Systems"* Paper Contribution

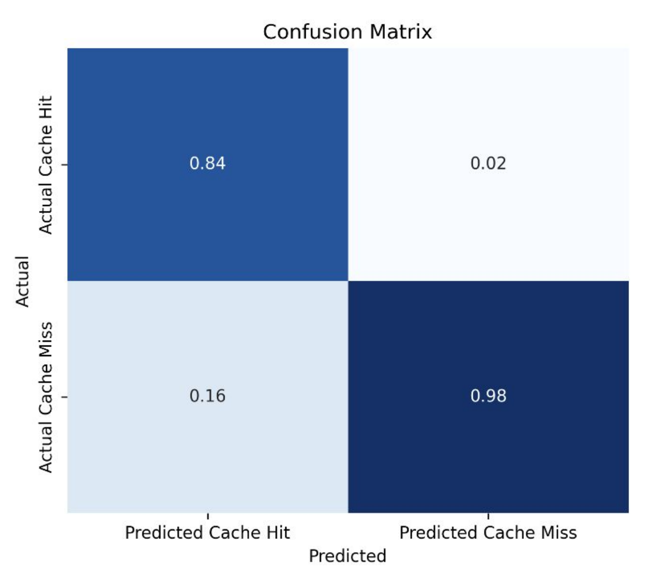
sentence space diagram

Contribution: Verifying Semantic Cache Timing Attack

✅Attack queries that are *cache hits* have a distinct response time difference compared to *cache misses*



✅When using an attack sentence representative of a large sentence space, an attacker can usually successfully retrieve the victim's answer



Contribution: Verifying Semantic Cache Timing Attack

Victim prompt: Schedule a meeting for the interdisciplinary team to review and discuss treatment options for **Ana** with **Poisoning**

Successful Attack: Design an agenda for the interdisciplinary team meeting to discuss and optimize the treatment plan for **Ana** with **Poisoning**

Failed Attack: Arrange an agenda for the interdisciplinary team meeting to brainstorm creative solutions for the challenges in treating **Ana** with **Poisoning**

Existing Threat Model

Unveiling Timing Side Channels **paper proposes the following threat model:**

- Attacker has knowledge of all prompts in the search space
- Black box access to model
- Can only query the model on the same platform as potential victims

BUT, there are some possible issues with this threat model/their implementation

- Very large search spaces to search through
- The authors only tested for when the attacker already *knows* a victim's PII, but not the rest of the prompt