CSE406: Computer Security Sessional

Side-Channel Attack

ID: 2005089

1. Introduction

This project demonstrates a website fingerprinting attack by using cache side-channel data collected via JavaScript. The goal is to infer which website is being visited in a browser tab, even if the attacker only has access to another tab. This is achieved by collecting cache access latency traces and training a machine learning model to classify the visited websites based on those traces.

2. System Design Overview

The system comprises:

- **Frontend (JavaScript)**: Captures timing traces using `performance.now()` and sends them to the Flask backend.

- **Backend (Flask, Python)**: Receives and stores traces, manages user interface, and serves training/testing functionality.

- **Trace Collection (Selenium)**: Automates the collection of traces by visiting selected websites and recording background activity.

- **Database (SQLite)**: Used to store traces persistently and keep track of how many traces have been collected per website.

- **Machine Learning (PyTorch)**: Includes both simple and complex 1D CNNs to classify the collected traces.

3. Task-1: Measuring Cache Latency

In the first task, we measured the memory access latency across varying buffer sizes to observe cache behavior. A JavaScript-based warm-up script was used to access increasing memory sizes and measure the median access latency over a 10-second interval, sampled every 10ms.

This latency data helps us understand the transitions across L1, L2, L3 caches and main memory — which is crucial for designing effective side-channel attacks using cache timing differences.

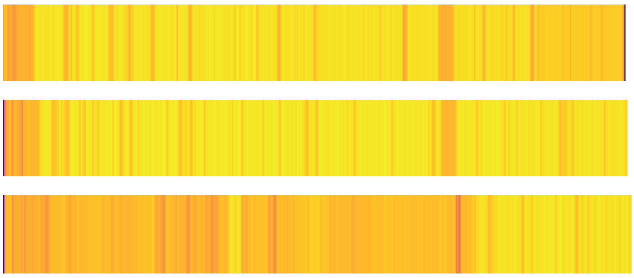
N	Median Access Latency (ms)
1	0.00
10	0.00
100	0.00
1000	0.00
10000	0.00
100000	0.20
1000000	0.70
10000000	4.70

The results show that up to around 10,000 elements, the latency remains negligible — indicating accesses are hitting L1/L2 cache. Beyond that, latency increases gradually, peaking significantly at 10 million, likely due to main memory accesses.

4. Data Collection

Traces were collected from three websites:

- `https://cse.buet.ac.bd/moodle/`
- `https://google.com`
- `https://prothomalo.com`



Each site had 17,000 traces (a total of 51,000 traces) after being combined collected in total from different environments. For improved generalizability, trace data from multiple machines were merged and **normalized per-user**, ensuring that range differences between systems did not bias the model. However, there will still be some noise in the data because of various reasons like background processes, browser rendering delays, CPU scheduling variance, network jitter, and occasional interference from unrelated system activity.

5. Data Normalization

To account for hardware differences across contributors, each contributor’s dataset was normalized using min-max normalization per trace. This ensures all data are on a consistent [0,1] scale before merging, which helps the model learn structure rather than absolute timings.

6. Experimentation Results

To evaluate the effectiveness of cache-based side-channel website fingerprinting, two convolutional neural network architectures—`simple_cnn` and `complex_cnn`—were trained and tested using both personal trace data (3000 samples) and the combined dataset (51,000 samples).

Results from Personal Trace Data (3000 samples)

Simple CNN:

- □ **Accuracy:** 91.87%
- □ **Best Performing Site:** *prothomalo.com* with 0.99 F1-score

Classification Report:

Website	**Precision**	**Recall**	**f1-score**
https://cse.buet.ac.bd/moodle/	0.88	0.91	0.89
https://google.com	0.91	0.85	0.88
https://prothomalo.com	0.97	1.00	0.99
Calculations			
Accuracy			0.92
Macro Average	0.92	0.92	0.92
Weighted Average	0.92	0.92	0.92

Complex CNN:

- **Accuracy:** 98.47%

- **Performance:** All three classes scored very high (≥ 0.96) in F1-score, showing strong generalization with minimal overfitting.

Classification Report:

Website	**Precision**	**Recall**	**f1-score**
https://cse.buet.ac.bd/moodle/	0.98	0.99	0.98
https://google.com	0.99	0.96	0.98
https://prothomalo.com	0.99	1.00	1.00
Calculations			
Accuracy			0.98
Macro Average	0.99	0.98	0.98
Weighted Average	0.99	0.98	0.98

These results show that, with clean and consistent data collected from a controlled environment, even a small dataset can yield high accuracy. The `complex_cnn` model especially demonstrated excellent performance due to its deeper architecture and ability to extract fine-grained patterns.

Results from Combined Dataset (51,000 samples)

Simple CNN:

- **Accuracy:** 76.56%
- **Observations:**
  - *prothomalo.com* again achieved the highest F1-score (0.86).
  - Performance for *moodle* and *google* dropped slightly due to noise from multiple sources.

Classification Report:

Website	**Precision**	**Recall**	**f1-score**
https://google.com	0.72	0.70	0.71
https://prothomalo.com	0.85	0.86	0.86
https://cse.buet.ac.bd/moodle/	0.72	0.73	0.73
Calculations			
Accuracy			0.77
Macro Average	0.77	0.77	0.77
Weighted Average	0.77	0.77	0.77

Complex CNN:

- **Accuracy:** 79.35%
- **Improved Stability Across Sites:**
  - Higher precision and recall across all three websites.
  - Indicates better generalization over noisy, heterogeneous data.

Classification Report:

Website	**Precision**	**Recall**	**f1-score**
https://google.com	0.72	0.76	0.74
https://prothomalo.com	0.89	0.89	0.89
https://cse.buet.ac.bd/moodle/	0.77	0.73	0.75
Calculations			
Accuracy			0.79
Macro Average	0.79	0.79	0.79
Weighted Average	0.79	0.79	0.79

7. Discussion

The experiment demonstrates that cache-based side-channel attacks can effectively identify visited websites. When using only my own normalized traces, both models performed well—**simple CNN** reached **91.9%**, and **complex CNN** achieved **98.5%** accuracy.

After merging 51,000 traces from different environments, performance dropped slightly (**76.6%** for simple CNN, **79.4%** for complex CNN), likely due to system and interaction variability. Still, the results confirm that even noisy, cross-device data can reveal distinguishable patterns, making website fingerprinting a feasible attack vector.