Timing Attacks in the Modern Web

Student: Ana Dumitras, Supervisor: Dr. Elisabeth Oswald, Project Type: research

University of Bristol, Department of Computer Science

Introduction

Side Channel Attacks are a powerful way of extracting information hiding inside a device by analysing the physical signals like power consumption, heat emission or time elapsed while performing an interesting operation. Timing attacks are one of the oldest types of side-channel attacks, where the time required to perform a certain operation is used to reveal private information about the system being attacked.

Browser timing attacks are a particularly insidious type of timing attacks as they allow the adversary to easily trick the victim into exposing meaningful information by merely accessing a website. Recently Oren et al.[a] proved that using such a technique the attacker can determine what websites users are accessing. Another important series of attacks was discovered by Van Goethem et al [b]. where they derive meaningful information from the amount of data an user has access to.

[a]Oren, Yossef, et al. "The Spy in the SandboxPractical Cache Attacks in Javascript."

[b]Van Goethem, Tom, Wouter Joosen, and Nick Nikiforakis. "The Clock is Still Ticking: Timing Attacks in the Modern Web." Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM, 2015.

1. JavaScript Cache Attacks

Oren et al. presented a method that enabled an adversary to spy on what the victim was doing while accessing an untrusted web page. They were able to distinguish between the top ten most accessed websites. The technique makes use of the JavaScript HighResolution Timing API to measure the time it takes to access a set of variables in order to determine if they were stored in cache or RAM memory. Up until Firefox 41, Safari 9 and Chrome 45, the JavaScript Timing API returned submillisecond time measurements. Recent versions of the most popular browsers no longer offer precise timing preventing this type of attack. Although the attack has been fixed in JavaScript, attacks are still possible as long as the adversary uses another client side scripting language.

Figure: Using JavaScript Timing API in Firefox 40

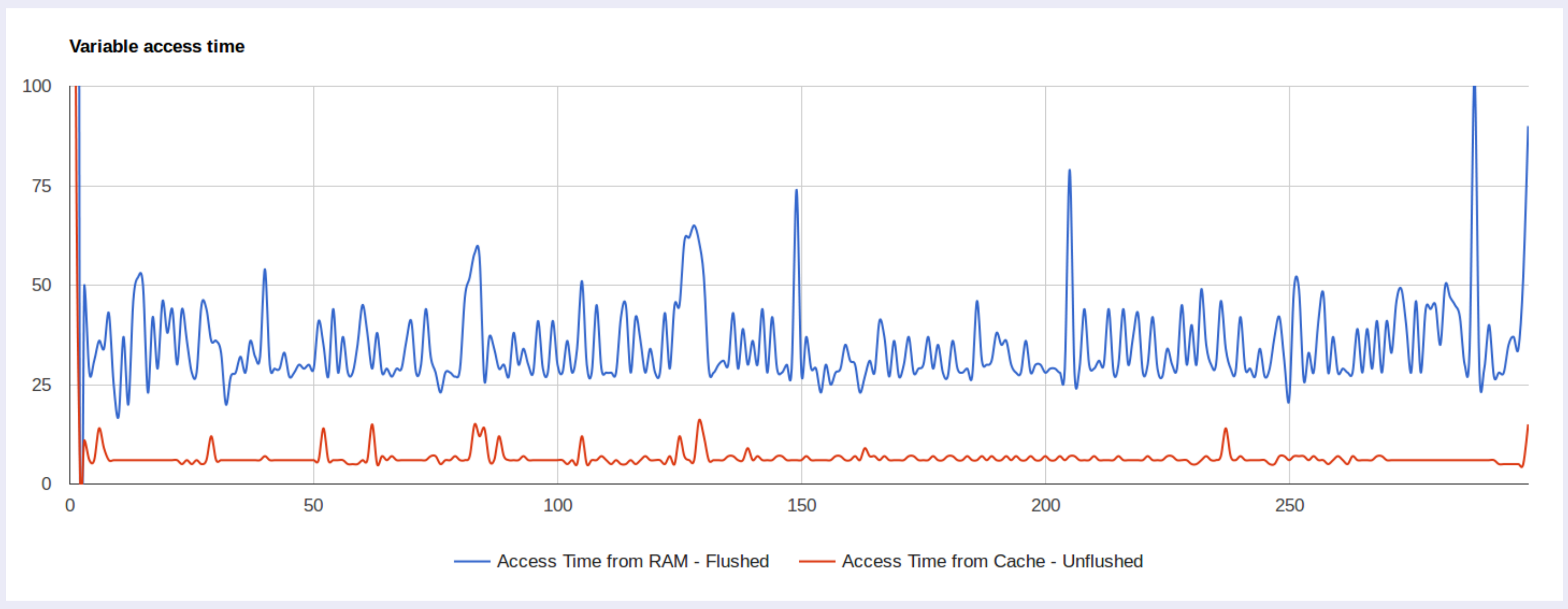
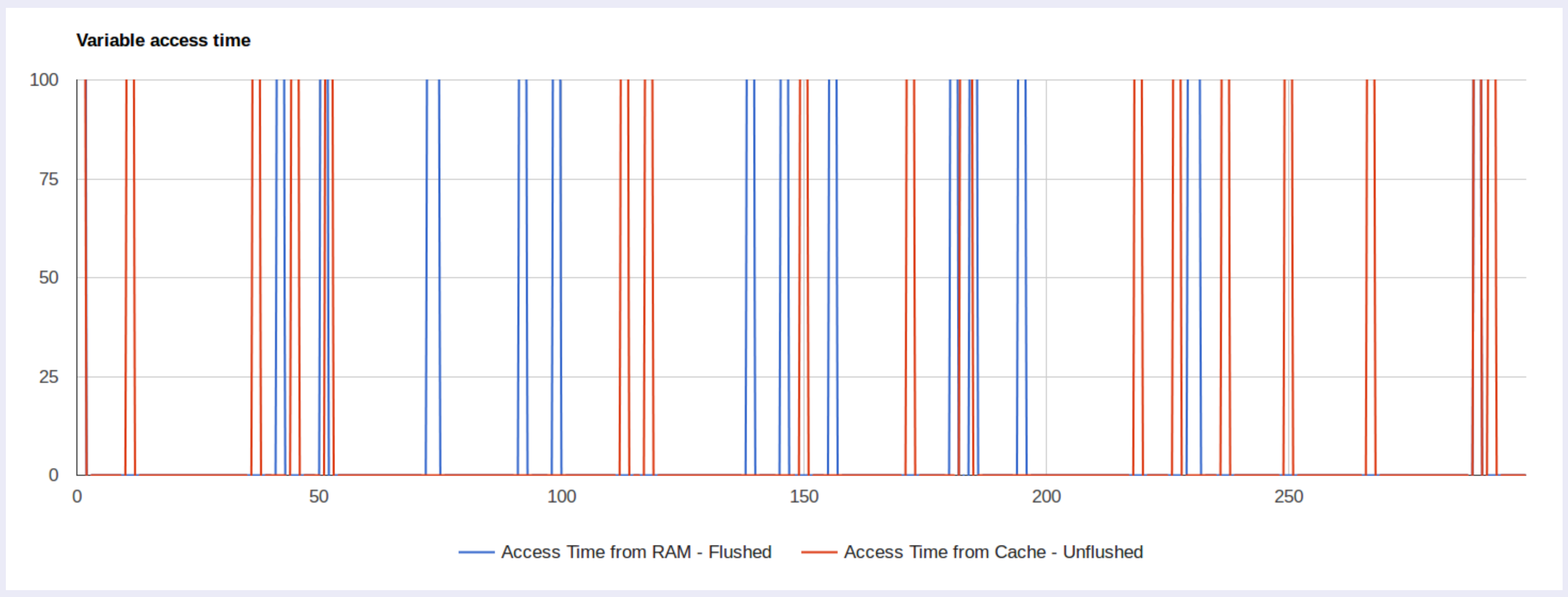
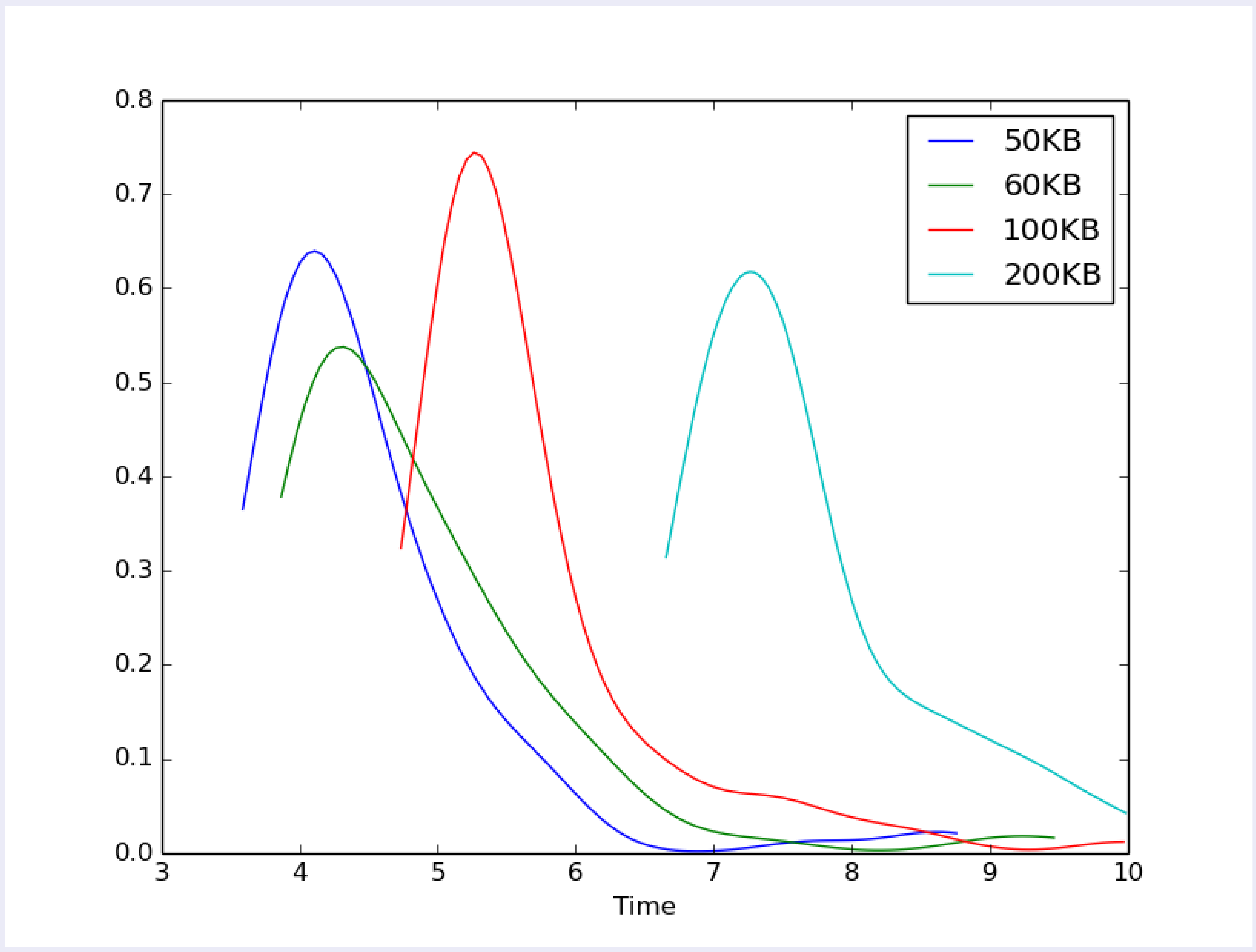


Figure: Using JavaScript Timing API in Firefox 41



2. HTML Attacks

HTML5 introduced two new elements: $<$ *audio* $>$ and $<$ *video* $>$. Once a resource is fetched, the browser will parse its contents in order to make it available for playing. The time it takes to parse a resource is dependent on the size of the file. The figure below shows that we can distinguish between the time it takes to process 4 files each of different sizes (50KB, 60KB, 100KB, 200KB).



HTML provided various method of determining the size of a file based on the time it takes to load it:

- Video parsing
- Application Cache
- Service Workers
- Script parsing

All these attacks are independent of the platform used to access the web site and affect a wide range of browsers, operating systems and devices.

3. Next Steps

- Implement the attack developed by Oren et al.
- Build a better classifier based on this attack.
- Evaluate new browser timing attacks.
- Use the data obtained from Van Goethem et al. attack to derive meaningful information about the victim.

4. Research proposal

Browser timing attacks can be prevented by improving

- the browser software (Chrome, Safari, Firefox)
- web development technologies (JavaScript, HTML, CSS, etc)
- cache - the JavaScript attack is not possible in CPUs with exclusive L3 cache