RUHR-UNIVERSITÄT BOCHUM

RUB

RUHR-UNIVERSITÄT BOCHUM

Spook.js: Attacking Chrome Strict Site Isolation via Speculative Execution

Exposee – November 20, 2021.
Chair for Network and Data Security.

Supervisor:	Prof. Dr. Jörg Schwenk
Advisor:	. . .

hgi Lehrstuhl für Netz- und Datensicherheit

1 Introduction

In this seminar thesis, I will first give a self-contained overview of the foundations of software-exploitable microarchitectural flaws in common consumer hardware. Based on the paper **Spook.js: Attacking Chrome Strict Site Isolation via Speculative Execution** [3] and the Proof of Concept code [2] released with it, I will then evaluate the risks of browser-based attacks on such flaws. Furthermore, I plan to present some of the mitigations put in place after disclosure, and critically discuss the real-world impact of the presented attack for end users.

1.1 Motivation

In recent years, web browsers have become increasingly important tools in our everyday digital interactions. Rather than through native applications for a plethora of platforms, digital services are accessed more than ever before using cross-platform web applications running in the browser. Hence, web browsers are becoming ever more complex. Furthermore, users place significant trust in browser security as such software has to protect a variety of information, from browsing history to passwords and banking details, against malicious web sites and their operators.

With the discovery of architecture-level vulnerabilities in processors of multiple major vendors, such as the Spectre vulnerability discovered in early 2018 [8], browser developers had to take on the task of defending against attacks that exploit flaws present far outside their own codebase [6]. While such efforts are ongoing, the Spook.js authors demonstrate that more work still needs to be done: Using JavaScript, the Spook.js attack on Google Chrome allows a malicious web site to exfiltrate arbitrary information from another site residing in the same rendering process. While Chrome attempts to isolate different sites into different processes [11], the authors argue that the isolation applied by Chrome is insufficient when user content is hosted on subdomains. Hence, Spook.js remains a powerful attack.

Given that such microarchitectural flaws blurring the distinction between software and hardware exploits have only recently been discovered, the IT security community can expect further research to uncover more exploits of the same kind. The existence of such exploits significantly alters the threat model that software security operates under: It is unsafe to think of hardware as fault-free and ultimately trustworthy. Strikingly though, the reported impact of such vulnerabilities seems to be rather low, as mentions of microarchitectural exploits in the wild are scarce. Hence, it will be worthwhile to assemble a concise overview of the modus operandi of this class of vulnerability, as well as to analyze the impact of microarchitectural exploits for the end user and encourage further research in that area.

1.2 Related Work

Research on the software-based exploitability of vulnerabilities in CPU designs has been ongoing for about two decades. A particularly noteworthy publication is that by D.J. Bernstein in 2004 on cache-timing attacks on AES [4]. Interest in software-exploitable physical side-channels on major CPU architectures then increased rapidly with the independent discovery of the Spectre [8] and Meltdown [9] vulnerabilities by several security research groups in 2018. Both exploit performance-improving techniques, such as out-of-order execution and speculative execution, found in modern desktop and server CPU generations.

The original Spectre attack, which is the basis for the attack described in the present paper, has since been further developed. Several new variants [5, 7, 10, 12] have been described in the literature, including vulnerabilities that are exploitable remotely or allow memory write access.

It should also be noted that Strict Site Isolation is not a concept introduced in the light of Spectre and its variants. Opportunities for security improvements to multi-process website rendering were documented as early as 2005 [1], providing the Chromium project and its derivatives with a large head start in defending the browser against Spectre-class vulnerabilities.

2 Work Packages

2.1 Overview

After submitting this research proposal, I will work mainly on three different focus areas:

- Assembling easily understandable technical background information on what aspects of the x86 architecture allow physical side-channel attacks to function
- Analyzing in more detail the provided Proof of Concept code and relating the findings to the present paper, then – time permitting – researching a timeline of mitigation efforts
- Giving a perspective on the real-world impact of architecture-level flaws

2.2 Detailed Description

Technical Background. In this initial work package, I will broaden the literature review done in preparation for this research proposal in order to provide a self-contained introduction to microarchitectural flaws on x86 systems. Based on the existing literature, I will then produce an initial version of the Technical Background section. Given the complexity of such attacks and their recent discovery, I expect this phase to take up around half of the available time frame.

Proof of Concept Evaluation. With the foundations of this seminar laid out, I will subsequently work with the Proof of Concept code provided by the Spook.js authors. First, I will attempt to reproduce the exploit on a local machine. Thereafter, I will attempt to reconstruct a coarse timeline of mitigations introduced into Google Chrome. Time permitting, I would like to gather more data on the influence of different hardware and software configurations on the viability of the exploit. I expect this phase to be finished around one week before submission of the initial draft.

Real World Impact. Based on the findings of the previous two work packages, I will critically discuss the real-world impact of the present and similar flaws for end users and identify areas where further research is required.

3 Organization of the Seminar Thesis

Tentatively, I plan on laying out the paper as follows:

Bibliography

[1] Site isolation design document – the Chromium projects, no date. URL `https://www.chromium.org/developers/design-documents/site-isolation`.

[2] Ayush Agarwal, Sioli O'Connell, Jason Kim, Shaked Yehezkel, Daniel Genkin, Eyal Ronen, and Yuval Yarom. spookjs/spookjs-poc: Proof-of-concept for Spook.js, 2021. URL `https://github.com/spookjs/spookjs-poc`.

[3] Ayush Agarwal, Sioli O'Connell, Jason Kim, Shaked Yehezkel, Daniel Genkin, Eyal Ronen, and Yuval Yarom. Spook.js: Attacking chrome strict site isolation via speculative execution. In 43rd IEEE Symposium on Security and Privacy (S&P'22), 2022.

[4] Daniel J. Bernstein. Cache-timing attacks on AES. 2005.

[5] Intel Corporation. INTEL-SA-00145, 2018. URL `https://www.intel.com/content/www/us/en/security-center/advisory/intel-sa-00145.html`.

[6] Mike West et al. Chromium Blog: mitigating side-channel attacks, 2021. URL `https://blog.chromium.org/2021/03/mitigating-side-channel-attacks.html`.

[7] Vladimir Kiriansky and Carl Waldspurger. Speculative buffer overflows: Attacks and defenses. 2018.

[8] Paul Kocher, Jann Horn, Anders Fogh, , Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. Spectre attacks: Exploiting speculative execution. In 40th IEEE Symposium on Security and Privacy (S&P'19), 2019.

[9] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. Meltdown: Reading kernel memory from user space. In 27th USENIX Security Symposium (USENIX Security 18), 2018.

[10] Giorgi Maisuradze and Christian Rossow. Ret2spec: Speculative execution using return stack buffers. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, page 2109–2122, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356930. doi: 10.1145/3243734.3243761. URL `https://doi.org/10.1145/3243734.3243761`.

[11] Charlie Reis. Google Security Blog: mitigating Spectre with site isolation in Chrome, 2018. URL `https://security.googleblog.com/2018/07/mitigating-spectre-with-site-isolation.html`.

[12] Michael Schwarz, Martin Schwarzl, Moritz Lipp, Jon Masters, and Daniel Gruß. NetSpectre: Read arbitrary memory over network. In Computer Security - ESORICS 2019, volume 1 of Lecture Notes in Computer Science, pages 279–299. Springer, September 2019. ISBN 978-3-030-29958-3. doi: 10.1007/978-3-030-29959-0_14.